National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, tests publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

NL-339 (r. 88/04)

Canada

THE UNIVERSITY OF ALBERTA

ALGORITHMS FOR GLOBAL ROUTING

by

© Ravibala Singh

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF Master of Science

Department of Electrical Engineering

EDMONTON, ALBERTA

Spring 1988

Permission has been granted to the National Library of Canada to microfilm this thesis and to lend or sell copies of the film.

The author (copyright owner) has reserved other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without his/her written permission.

L'autorisation a été accordée à la Bibliothèque nationale du Canada de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur (titulaire du droit d'auteur) se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation écrite.

ISBN 0-315-42734-5

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR Ravibala Singh

TITLE OF THESIS ALGORITHMS FOR GLOBAL ROUTING

DEGREE FOR WHICH THESIS WAS PRESENTED Master of Science

YEAR THIS DEGREE GRANTED 1988

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(SIGNED) R.A.Singh

PERMANENT ADDRESS:

Flat F-5B

SAKET, New Delhi 110017

India

DATED 2nd March 1988

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled ALGORITHMS FOR GLOBAL ROUTING submitted by Ravibala Singh in partial fulfilment of the requirements for the degree of Master of Science.

Jack Macbeth

Supervisor

Date March 1, 1978

Abstract

This thesis presents a global router for standard-cell ICs that is now part of a standard-cell design system developed at the University of Alberta. The global router incorporates a new approach to the global routing problem in standard-cell layout. Traditional approaches often transform the problem into a graph theoretic one and tend to ignore much of the physical information of the routing environment that could be gainfully used to obtain better routing solutions. Based on an analysis of net configurations of standard-cell circuits, two sub-problems that arise in the global routing of the circuits have been identified. The new global router adopts a two phase approach that focuses on these sub-problems and attempts to deal with them effectively. Experiments indicate that, as compared with the conventional minimum spanning tree global router, the new global router global-routes a set of 15 test circuits with about 4% fewer tracks.

Acknowledgements

I would like to thank my thesis supervisor, Dr. J.T. Mowchenko, for introducing me to the subject and acknowledge with gratitude his constant support and encouragement throughout the course of this work. I would also like to thank the members of my examining committee, Dr. E.F. Girczyc and Dr. J. Schaeffer for their helpful suggestions and comments. Special thanks go to Mr. Raymond Ma for his invaluable assistance and advice and to my collegues in the Electrical Engineering department for all the help recieved. I would also like to acknowledge the encouragement and extensive support recieved from my family.

Finally, I wish to thank my supervisor and the Department of Electrical Engineering for the financial support recieved during the course of this work.

Table of Contents

Chapter		Page
1.	Introduction	1
	1.1 Topic of the Thesis	6
	1.2 Organization of the Thesis	7
2.	Background	8
	2.1 Routing - some definitions and terms	8
	2.2 Routing Algorithms - a classification and discussion	11
	2.3 Standard-Cell Routing	16
3.	Global Routing in Standard-cell Layouts	20
	3.1 Problem Formulation	20
	3.2 Previous Work	25
4.	Algorithms for Global Routing	36
	4.1 Motivation for the Approach taken to the Problem	36
	4.2 A Two-Phase approach to Standard-Cell Global Routing	40
	4.3 Algorithms for Net-segment Selection	41
	4.3.1 Algorithm A	41
	4.3.2 Algorithm B	45
	4.3.3 The Select-Right Algorithm	47
	4.4 An Algorithm for Assigning Switchable Net-Segments to Channels	49
	4.4.1 Motivation for the Design of the Algorithm	50
	4.4.2 Simulated Annealing	51
	4.4.3 Some Observations	53
	4.4.4 A PHC Algorithm for the Assignment of Channels to Switchable Net-segments	57
5.	Experiments and Evaluation	61

5.1 Implementation and Environment for Evaluation ...61

5.2 Experiments with the Phase I Algorithms 62

5.2.1 Performance Comparison of the Algorithms for Net-segment Selection 63

5.2.2 Experiments with Channel Ordering and FMAX 66

5.3 Experiments with the Phase II Algorithm 67

5.4 Evaluation of the Two-Phase Global Router 71

6. Conclusions and Directions for Future Work 86

Bibliography 90

Appendix I - A Simulated Annealing Algorithm for Assigning Switchable Net-segments to Channels 94

Appendix II - A Minimum-Spanning-Tree Global Router 96

List of Tables

Table		Page
5.1	Statistics of the test circuits	75
5.2	Experimental results obtained after Phases I and II using Select-Right, Algorithm A and Algorithm B as the Phase I algorithm for the Two-Phase Global Router	76
5.3	Experimental results obtained by varying the order in which channels are processed	77
5.4	Experimental results comparing the SHCH algorithm with the Simulated Annealing algorithm, both algorithms being allowed the same CPU time per instance	78
5.5	Experimental results obtained using the Simulated Annealing algorithm	79
5.6	Experimental results obtained with the Two-Phase Global Router, Track-Filling, Minimum Spanning Tree, and the Left-Edge algorithms, and with no global routing	80

List of Figures

Figure		Page
1.1	A standard-cell layout	3
1.2	Typical gate-array and general-cell layouts	4
2.1	Euclidean/Manhattan distance	9
2.2	A two layer Manhattan routing	10
2.3	Wiring on a grid	10
2.4	Establishing inter-channel connections	17
3.1	Standard-cell layout model	21
3.2	Construction of graph, G_n	23
3.3	An example of a global routing problem	24
3.4	Implementation of a net-segment	25
3.5	A standard-cell layout model	26
3.6	Classification of net-segments	30
3.7	Ranking of net-segments	33
3.8	Choosing between a non-essential net-segment and a non-switchable net-segment	34
4.1	Net configurations for 2-terminal nets	37
4.2	Net configurations for 3-terminal nets	38
4.3	Algorithm A	43
4.4	Algorithm B	46
4.5	The Select-Right algorithm	48
4.6	Rules for net-segment selection	49
4.7	The basic Simulated Annealing algorithm	54
4.8	The SHCH Algorithm - for Phase II	58
4.9	Generating a perturbation	58
4.10	Results of a perturbation	60

Figure Page

5.1 Choosing between a non-essential net-segment and a non-switchable net-segment 64

5.2 log_{10}(total number of net-segments) vs log_{10}(time) for Algorithm A 81

5.3 log_{10}(total number of net-segments) vs log_{10}(time) for Algorithm B 82

5.4 log_{10}(total number of terminals) vs log_{10}(time) for the Select-Right algorithm 83

5.5 Initial and final density profiles 84

5.6 Density profiles after Phases I and II 85

6.1 Moving a non-switchable net-segment 88

6.2 Detouring of a net-segment 89

Chapter 1

Introduction

Semiconductor technology has evolved very rapidly over the last two decades. The number of transistors that can be economically integrated on a single chip has doubled every one to two years over this period. The projected improvements in fabrication technology will make chips containing several million transistors possible within the next few years. With this 'Very Large Scale Integration' (VLSI) of transistors on a chip, entire systems can now be fabricated on a single chip. In order to cope with the enormous complexities involved in the design of such chips, computer-aided design (CAD) tools are used at every step of the design process.

One step in the integrated circuit (IC) design process which uses CAD extensively is the layout of the IC. The layout of a circuit refers to the process of placing the components of the circuit on a chip and connecting them by wires according to a given set of rules, called design rules. The circuit layout phase is important because it determines the chip area, which has a major influence on chip costs. It, therefore commands a substantial portion of the chip design time.

Circuit design methodologies in current use can be broadly classified into two categories: full-custom and

semi-custom. Full-custom ICs are designed and laid out manually at the transistor level. With the full-custom approach, a designer can obtain high packing density and high performance at the cost of increased design effort and time. Because of the high design costs, a full-custom design is only used for circuits which are produced in high volume, or where performance optimization and area minimization outweigh design costs.

The alternative to full-custom design is semi-custom design. Semi-custom ICs are mainly designed and laid in the **cell-based** design styles. The term 'cell' refers to a pre-defined functional unit, such as a NAND gate. A design is specified in terms of these cells and their interconnections. A major advantage of using cell-based designs is that the layout can be produced by highly automated software. As a result, development time and costs for semi-custom ICs are lower than for full-custom ICs. The penalty paid is loss of flexibility and lower packing densities.

There are three major cell-based design styles: standard-cell (also called polycell), gate-array, and general-cell (or macro-cell). Since this thesis work deals with the standard-cell design style, it is explained in some detail followed by a brief description of the other two styles.

The term **standard-cell** was originally an abbreviation for 'standard height cell'. Each standard-cell is a

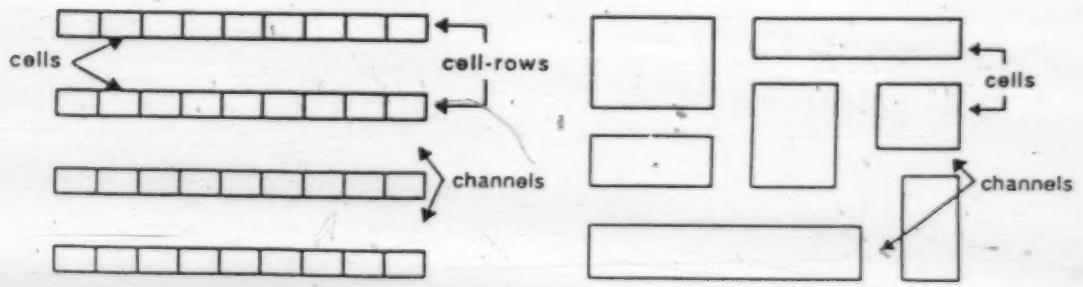
functional unit which has been pre-designed and laid out manually. The designer is provided with a library of standard-cells which contains the complete mask layout of the cells. The cells are of rectangular shape, generally with standard heights and varying widths. The cell terminals lie along its top and/or bottom sides. A standard-cell IC is laid cut by placing the cells in rows and running the cell interconnections in the channels between the rows. Figure 1.1 shows a typical standard-cell layout.

A gate-array is a regular array of identical cells which has been pre-fabricated on a chip upto but excluding the interconnect and contact layers. Each cell is usually a simple gate or a small group of unconnected transistors. The gate-array is customized to the user's needs by specifying one or more layers of metal to interconnect the cells and thus implement the circuit on the array.

Figure 1.1 A standard-cell layout

In the general-cell approach, the cells implement more complex functions and are physically larger than standard-cells. Further, there is no restriction on the physical size or shape of a general-cell, though they are generally rectangular. Figures 1.2 (a) and (b) show typical gate-array and general-cell layouts respectively.

Each of the cell-based design styles has its advantages and disadvantages. In the standard-cell approach, high-performance, special-purpose functions can be implemented by creating new standard-cells. Because cells are used more than once, the time spent on area and performance optimization will be amortized over many designs. Unlike gate-arrays, however, there is no saving in fabrication time, since the complete fabrication process must be carried out. Gate-array ICs, on the other hand, do not use chip area as efficiently as the other cell-based



(a) A gate-array layout (b) A general-cell layout

Figure 1.2 Typical gate-array and general-cell layouts

styles. Sometimes, transistors within cells go unused or entire cells remain unused because the fixed width channels reserved for wiring may not be able to accomodate all the interconnections required. Unlike standard-cells and gate-arrays, general-cells can implement much more complex functions. However, as the functions they implement become more complex and special-purpose, they lose the flexibility in being able to be used in many circuits. Also, the automatic layout of general-cell ICs is much more difficult. All these factors must be considered before choosing one cell-based style over the others.

During circuit layout, all the above design styles involve an optimal **placement** of the cells and the interconnection, or **routing**, of the cells to form the final circuit. The objective of the placement procedure is to assign specific locations to the cells so that they can be interconnected in a compact fashion in the routing phase. The routing procedure determines the actual wiring paths which will interconnect the cell terminals in conformance with the circuit design. The objective of routing is to complete the required interconnections in minimum area without violating any design rules. Because of the inherent complexity of the routing task, it is sometimes performed in two steps. The first step, called **global routing**, involves the assignment of the wiring paths to the channels. During the second step, called **detailed routing**, the exact locations of the wiring paths within each channel are

determined.

1.1 Topic of the Thesis

A standard-cell design system is being developed at the University of Albeta, to be used as a research tool for developing and testing layout algorithms. The system runs on the VAX 11/780 computer under UNIX' 4.3 BSD and is implemented in the *C* programming language with over 25,000 lines of source code. There are four major components in this system: a cell library, a circuit editor, placement programs and routing programs. This thesis reports the research work done on the global routing phase in standard-cell routing.

The major result presented in this thesis work is a global router that incorporates a new approach to the global routing problem. Most of the existing global routers use a graph theoretic approach. One limitation of this approach is that the physical information of the routing environment and the restrictions imposed by it are not adequately made use of. Based on an analysis of net configurations of actual standard-cell circuits, the proposed global router adopts a two phase approach. Three algorithms are presented for the first phase. A *Probabilistic Hill Climbing* algorithm is presented for the second phase. Experiments indicate that as compared with the conventional minimum spanning tree global router, the new global router global-routes a set of 15 test

---

'UNIX is a trademark of Bell Laboratories

circuits with about 4% fewer tracks.

1.2 Organization of the Thesis

Chapter 2 discusses the routing problem and presents a review of the common approaches used by routing systems. It then proceeds to describe the routing environment in standard-cell ICs and the general objectives of a standard-cell routing system. Chapter 3 focuses on the global routing problem in standard-cell ICs providing a precise definition of the problem and a critical review of standard-cell global routers in current literature. Chapter 4 discusses the motivation for the proposed approach to the standard-cell global routing problem and describes the four algorithms that constitute the new global router. The global router has been implemented and the results of experiments conducted to evaluate its performance are discussed in Chapter 5. In Chapter 6, we summarize the main points of this research work and offer suggestions for future work.

Chapter 2

Background

Chapter 2 begins by introducing some concepts and terms associated with the routing of ICs in general. The routing problem is then defined and existing methods for wire routing reviewed. Since the thesis deals with routing of standard-cell ICs, we proceed to examine the routing requirements of standard-cell ICs in particular, discussing the routing environment and the general objective of the routing algorithms.

2.1 Routing - some definitions and terms

Routing is the last step in the circuit layout process. The placement procedure first places the components of the circuit on the chip. The routing process then proceeds to formally establish the wiring paths necessary to interconnect the components in order to realize the circuit. The interconnections between the components is specified by a set of nets,

$$N = \{n_1, n_2, \ldots, n_j\}$$

called the net-list. Each net is a set of connections between the terminals of components. A net, therefore, is specified by a set of terminals,

$$n_i = \{t_1, t_2, \ldots, t_k\}$$

to be connected together.

The manner in which interconnect wires are run is governed by a set of rules called the wiring model. In general, wiring paths can be piecewise linear curves, however, in automated routing systems, wires are usually constrained to contain horizontal and vertical wire segments only. An immediate consequence of this approach is that the conventional Eucilidean distance formula no longer applies. Thus in Figure 2.1 the minimum wire distance between points A and B is (x+y) since diagonal runs are not permitted. This distance measure is called the rectilinear or Manhattan distance. A wiring model most commonly used in routing standard-cell ICs is the Manhattan wiring model. This model assumes rectilinear wiring with two layers available for interconnections, One layer is reserved for vertical segments and the other for horizontal segments. Where a vertical and horizontal segment intersect, a contact cut or via may be introduced to interconnect them. Figure 2.2 shows an example of wiring in the Manhattan model. The horizontal wire- segments (indicated by dashed lines) are assigned to one layer and the vertical segments are assigned to the

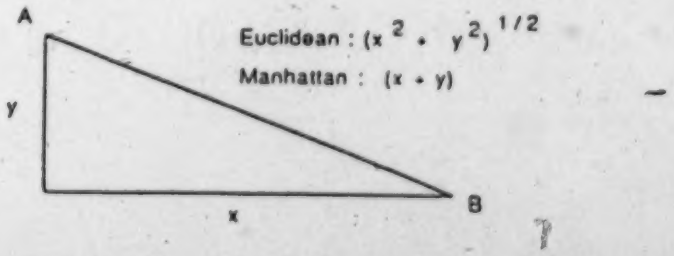


Figure 2.1 Euclidean/Manhattan distance

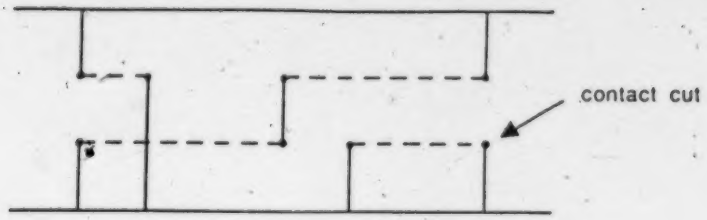


Figure 2.2 A two layer Manhattan routing

other layer.

Frequently, the positions of terminals and wires in a routing are restricted so that they lie on a grid consisting of horizontal lines called tracks and vertical lines called columns. The minimum separation between the grid lines permitted by the fabrication technology is referred to as the minimum inter-wire spacing. Each layer on which wiring is permitted may have its own wiring grid. Figure 2.3 shows an example of wiring on a grid. Vertical segments are assigned to Layer 1 and lie along the vertical grid lines on that layer. Horizontal segments lie along the horizontal

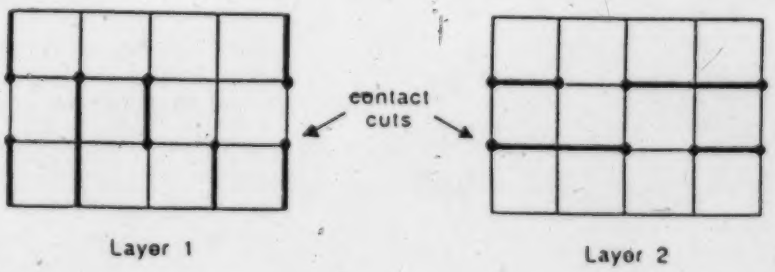


Figure 2.3 Wiring on a grid

grid lines in Layer 2. A contact cut is introduced to connect a vertical and horizontal segment.

Having defined some terms associated with routing of ICs, we now state the routing problem in its most general form as follows: Given a placement of the components of a circuit and a net-list, realize the interconnections specified by the net-list so as to minimize a given cost function. Typical cost functions are the area occupied by the routing, and the total wire length.

Most wire routing problems, except the highly restricted ones, are computationally intractable. Some of them have been shown to be NP-Complete [Lap80, Szy85, Joh82]. This implies that there are probably no computationally efficient algorithms that will give optimal solutions with respect to any reasonable cost function. Almost all routing algorithms are therefore heuristic algorithms which attempt to find reasonably good solutions in an acceptable amount of computation time. Several wire routing problems and their complexities are discussed in [LaP80]. The following section presents a classification and discussion of the commonly used routing algorithms

2.2 Routing Algorithms - a classification and discussion

Wire routing algorithms can be broadly classified as:

1. Grid expansion algorithms or 'maze-routers'.
2. Line expansion or 'line-probe' algorithms.
3. Channel algorithms.

The Lee router [Lee61] is perhaps the most widely used 'maze-router'. The router imposes a rectangular grid over the entire routing area. Terminals are represented by the cells of the grid (i.e. grid squares) in which they lie and wires are represented by a sequence of adjacent cells. Given a pair of terminals to be connected by a wire, one of the terminal cells is selected as the starting cell and the other as the target cell. Beginning at the starting cell, the maze-router builds a set of paths towards the target cell. At each step, the router extends all the current paths by one cell in all possible directions. Only cells which are not inside components or which are not already part of some path may be used. Because one path may be extended in several possible directions, the number of paths tends to grow after each step. The path extending process can be viewed as a wave that begins at the starting cell and gradually expands outwards towards the target cell. The expanding wavefront is circular with the starting cell at its center. The paths to the target cell follow the wavefront as it propagates on the rectangular grid. The process continues until either all the paths become blocked or the target cell is reached. The algorithm guarantees to find a path with a minimum cost, if one exists, regardless of how complicated the path may be. Path cost is any measure the user wishes to minimize and may include rectilinear distance, cross-overs with existing wires, penalties for entering congested areas, etc.

'Maze-routers' remain in widespread use because of their enormous flexibility and ability to route dense chips - they are even used with other routers to finish routings that other routers have failed to complete. The primary disadvantage of maze routers is that they are slow and require a large amount of memory. Approaches to improve speeds by giving priority to grid expansion in the direction of the target point [Rub74] have shown significant improvements.

'Line-probe' routers [Hig69, Mik68] are faster and much more efficient as compared to the Lee-type routers. The algorithm in [Hig69] finds a connection between two terminals by constructing a sequence of line segments emanating from each terminal. Line segments are constructed by starting from each of the two terminals and running horizontal or vertical lines towards the other terminal. Consider, for instance, that a vertical line is run from one of the two terminals, say the source terminal, towards the other, referred to here as the target terminal. If the vertical line is blocked by some obstruction, an 'escape' point is found on the vertical line, just short of the obstruction. The vertical line segment up to the escape point is added to the sequence of line segments emanating from the source terminal. A horizontal line is now run from the escape point towards the target terminal and another line segment is selected to be added to the sequence. A similar sequence of line segments is constructed with the

roles of the target terminal and source terminal reversed. The process continues until the two sequences intersect and a path is established. The above algorithm does not guarantee a connection even if one exists. 'Line-probe' routers usually fail to complete routings in complicated wiring problems.

An approach combining the line expansion and grid expansion techniques [Sou78] has the advantages of both methods. It is much faster than the Lee-type router and guarantees a connection if there is one. A line-search is first directed towards the target. When the line-search hits an obstacle, an expansion technique typical of the Lee-type algorithm is used to "bubble" around the obstacle. The line-search can then continue towards the target.

The third category of routing algorithms, the channel algorithms, were first proposed by Hashimoto and Stevens [Hash71]. Channel algorithms work by first partitioning the routing space into smaller regions or channels and then routing all the connections within each channel separately. A channel is a continuous area between the components of the placed circuit. Because components usually have rectilinear shapes, the routing space is generally partitioned into rectangular channels. After channels have been determined, channel algorithms use a two-step approach to perform routing. The two steps are:

(1) Global routing - In global routing (also called topological or loose routing), a rough wiring path for each

net is determined through the channels. The objective of this step is to come up with a general wiring strategy that distributes wiring congestion between channels and ensures that wire capacities are not exceeded in any region of the chip.

(2) Detailed routing - At the end of global routing, individual net-lists are generated for each channel, specifying the terminals and their interconnections in the channel. During detailed routing, each channel with its connections is considered independantly and the exact wiring path for each connection within the channel is finalized. Detailed routing of channels is also called **channel routing**. (Please note the distinction between the terms 'channel algorithm' and 'channel routing'. A channel algorithm is a general method of wire routing, consisting of two major steps, one of which is channel routing.)

Channel algorithms have been used extensively for routing of ICs for many years because they are fast, efficient and simple. Since the first channel router was introduced by Hashimoto and Stevens, the channel routing problem has been extensively studied and many channel routers have been developed [Yos82, Riv82]. An excellent survey of channel routers appears in [Bur85].

Since the focus of this thesis work is on the problem of routing in standard-cell ICs, the following section examines the particular routing requirements of standard-cell ICs.

2.3 Standard-Cell Routing

As mentioned earlier, the layout of a standard-cell circuit typically consists of individual cells placed end-to-end in horizontal rows. Standard-cells are rectangular and normally have the same height but varying widths, encouraging such a row-wise packing of cells.

Early standard-cells were designed with terminals on only one side of the cell. Standard-cells in current use have terminals on two opposite sides such that pairs of electrically equivalent terminals appear directly opposite one another on the top and bottom of the cell. We will refer to cells with such a terminal configuration as **double-entry** cells.

In general, a net connecting terminals at various locations on the chip may lie entirely within one row, span two adjacent rows, or it may span several rows. If a pair of terminals on a net are separated by a row that does not contain a terminal of that net, then a special '**feedthrough**' cell is inserted into that row. A feedthrough cell is essentially a wire which allows the net to pass through a row. For double-entry cells, equivalent terminals can be used to conveniently connect nets that span two channels thus reducing the need for feedthroughs as compared with systems using single-entry cells. Figure 2.4 illustrates the use of both equivalent terminals and feedthrough cells, to establish inter-channel connections.

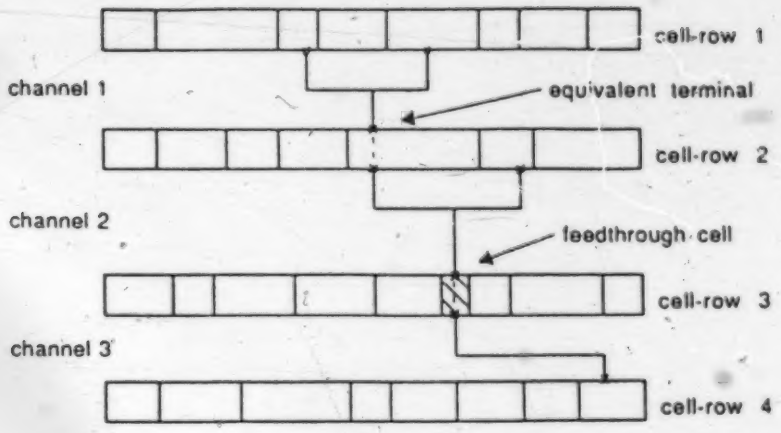


Figure 2.4 Establishing inter-channel connections

Though the general objective of a standard-cell routing system is to perform the interconnections using minimal routing area, it is imperative that circuit performance specifications are met. During routing, the major consideration in ensuring correct circuit performance are the signal delays in the interconnection wires. In designs where signal delays are critical, the routing procedure often incorporates timing information to bias the routing process. Most routers for standard-cell ICs use the Manhattan wiring model. Frequently, the horizontal wire segments in a channel are constrained to lie along horizontal grid lines called tracks. Because routing is restricted to the channels, the general objective of reducing the routing area translates to minimizing the total area occupied by the channels. Channel lengths are dictated

by the placement procedure, therefore, the objective of the router is to minimize the sum of the channel widths i.e. the total number of tracks used in the routing.

A lower bound on the width of a channel, or the number of tracks in a channel, is the channel density. The channel density is the maximum number of nets that are split by any vertical cut of the channel. A net is split by a vertical cut if there is at least one terminal of the net on either side of the cut or on the cut. In practice, channel density is also a reasonable upper bound on the channel width since most channel routers can route channels within one or two tracks of the channel density. Therefore, standard-cell routers generally use the sum of the channel densities for all the channels as the cost function to be minimized.

Unlike routing for gate-arrays, local wiring congestion does not pose a problem in standard-cell routing. In gate-array layouts, channel capacities are fixed sometimes making a 100% completion of routing impossible to achieve because of wiring congestion. By contrast, in standard-cell layouts, channel widths are allowed to vary so as to accomodate the needed routing tracks.

Since standard-cells are laid out in rows with intervening channels to contain the routing, most standard-cell layout systems employ channel algorithms to route them. The routing proceeds in two phases. During the global routing phase nets are assigned to channels. Exact wiring paths within channels are determined in the detailed

routing phase by a channel router. Since the aim of this research work is to develop better standard-cell global routers, we restrict our attention in subsequent chapters to global routing. In the next chapter, the global routing problem in standard-cell ICs will be formally defined and an overview of the literature on this specific topic will be presented.

Chapter 3

Global Routing in Standard-cell Layouts

As discussed in the previous chapter, the purpose of global routing is to provide a loose initial routing of the nets on a chip so as to guide the subsequent detailed wiring. This chapter focuses on the global routing problem in standard-cell ICs. The problem is formulated in terms of its graph theoretic model and a critical review of standard-cell global routers in current literature is presented.

3.1 Problem Formulation

We will first briefly discuss the layout model used by the U. of A. standard-cell design system so that we can formulate the global routing problem for this model.

The layout model used by the standard-cell design system is shown in Figure 3.1. Standard-cells of uniform height and varying widths are arranged end-to-end in horizontal rows by the placement phase, with intervening channels to contain the routing. The placement algorithm introduces feedthrough cells to connect terminals of nets that are separated by a row that does not contain a terminal of the net. This ensures that, for the routing phase, all nets have terminals on consecutive rows. The system uses double-entry cells, thus increasing the flexibility in

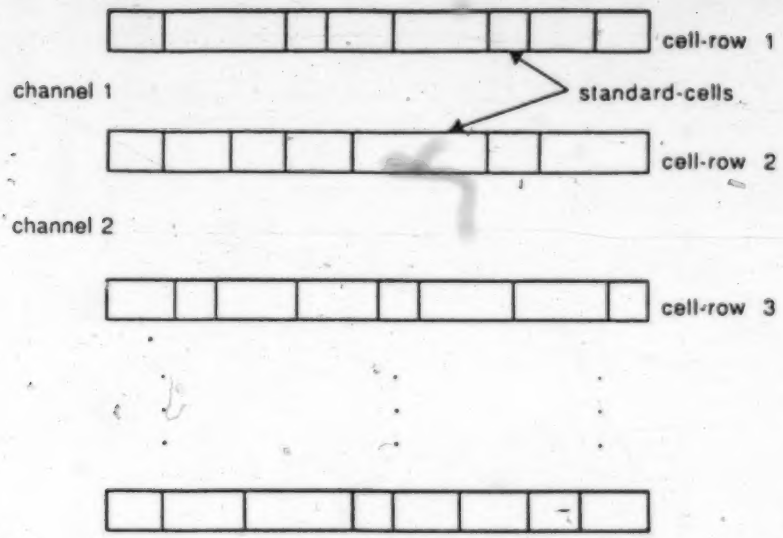


Figure 3.1 Standard-cell layout model

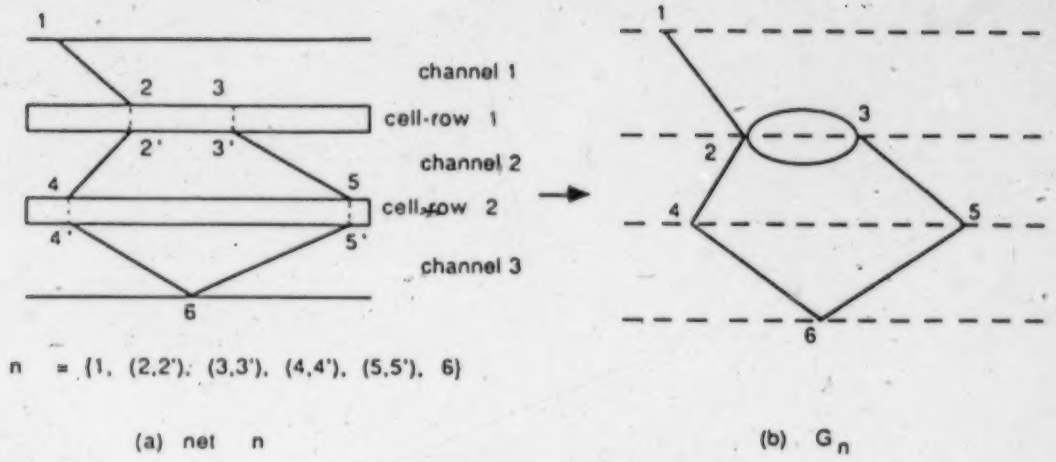
assigning nets to channels during the routing phase. The routing phase comprises of a global routing followed by a detailed routing using a channel router. The routers in the design system use the Manhattan wiring model described in Chapter 2.

We will not consider the connections to the I/O pads, therefore, routing is restricted to the channels and is not permitted beyond the cell rows. We also assume that the routing of the power and ground lines has been taken care of because standard-cells have internal horizontal power and ground buses which abut when the cells are placed in rows.

Having described the main features of the layout model used by the design system, we will now formulate the global routing problem for the model. The inputs to the global routing problem are a cell placement and a net-list of the

circuit. As mentioned earlier, each net n is a set of terminals, $n = \{t_i\}$ to be connected together. Consider a net n to be routed in the placed circuit. From the placement of the cells, the location of the terminals of n can be determined. In each channel that contains at least two terminals of n, we order the terminals by their x-position. Two terminals, t_j, $t_k \in n$, in a channel C, are said to be **adjacent** if, by the ordering, there is no intervening terminal from n in C. We will decompose n into a set of **net-segments** in each channel that contains at least two terminals of the net. A net-segment of n in a channel C is defined by an ordered pair of adjacent terminals, (t_j, t_k) from n in C. The horizontal interval in the channel between the two adjacent terminals is called the **span** of the net-segment. Let $S(n)$ be the set of all the net-segments on the chip for net n.

For net n, we will now introduce its graph model. Consider a graph, $G_n = (V, E)$ corresponding to the net n. Each vertex, $v_j \in V$ corresponds to a terminal $t_j \in n$. A pair of vertices v_j and v_k are connected by an edge in E iff $(t_j, t_k) \in S(n)$. Figure 3.2 shows a net and its corresponding graph representation. A pair of equivalent pins on a cell is represented by a single vertex. Since a graph with N vertices can always be connected by $N - 1$ edges which form a spanning tree covering all the vertices of the graph, an N-terminal net can always be connected up by $N - 1$ net-segments. A global routing of a standard-cell IC is,



n = {1, (2,2'), (3,3'), (4,4'), (5,5'), 6}

(a) net n (b) G_n

Figure 3.2 Construction of graph, G_n

therefore, equivalent to the spanning trees for the graph G_n for every net n. The edges which are on the spanning trees define the net-segments which make up the global routing. If a net-segment is part of a global routing, then the pair of terminals which define the net-segment must be connected in the segment's channel.

Let D_i denote the density of channel i after all the nets have been connected up by net-segments. We can now state the standard-cell global routing problem as follows:

Given a cell placement and a net-list of the circuit, for each net n in the net-list determine a set of net-segments which form a spanning tree covering n's terminals so as to minimize the sum of channel densities over all channels, ΣD_i.

To illustrate the global routing problem, Figures 3.3 (b) and (c) shows two ways to route net A and net B from

Figure 3.3 (a). Selecting the net-segments of Figure 3.3 (b) to connect up nets *A* and *B* results in a total channel density of 4 while selecting the set of net-segments in Figure 3.3 (c) results in a total of 2, and is therefore preferable.

A net-segment can be implemented with one horizontal wire and two vertical wires in a detailed routing (Figure 3.4). Depending on which track the horizontal wire segment is placed at the end of the detailed routing, the length of the two vertical wire segments will vary. Because the positions of the terminals are fixed, the length of the

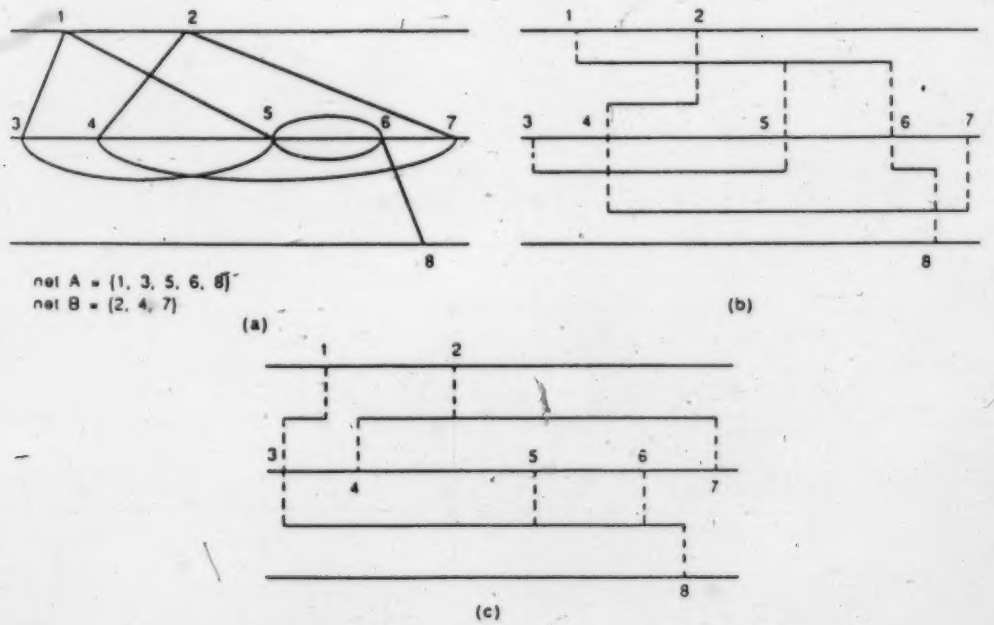


Figure 3.3 An example of a global routing problem

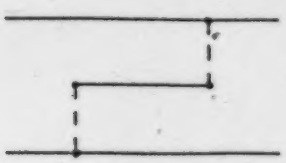


Figure 3.4 Implementation of a net-segment

horizontal wire segment is fixed. Since in global routing we are not concerned with the exact positions of the wires, it suffices to consider only the horizontal wire segment which implements a net-segment.

3.2 Previous Work

Much of the discussion in the literature on global routing has been directed at gate-array ICs [Aos83, Tin83] and general-cell ICs [Riv82, Kim83]. Few global routers for standard-cell ICs have been reported. This can be attributed to two reasons: (1) Early standard-cells were designed as single-entry cells, with terminals placed on one edge of the cell only. With such cells, the flexibility of assigning nets to different channels is limited and therefore not much emphasis was placed on global routing. (2) In a standard-cell layout, the channel widths can be increased so as to be able to complete the routing. Since a solution is always guaranteed, although it may not be the most efficient, a global router is not mandatory. Consequently,

some standard-cell layout systems do not employ a global router [Meh84]. Instead, a channel router is used to route the channels one at a time. With such a 'channel-at-a-time' approach the quality of the resulting routing is very much dependant on the order in which the channels are routed.

We will first review some global routers for standard-cell ICs which have been reported in the literature and then discuss in some detail a global router that is presently part of the standard-cell design system at the University of Alberta.

The Seimens 'Avesta' system [Kol77], uses single-entry cells which are arranged in rows placed back-to-back as shown in Figure 3.5. Two parallel rows with their interconnection channel running between them form a block. Such blocks are partitioned by vertical interconnection channels to form sub-blocks. For nets with segments in more than one sub-block, a rectilinear Steiner tree is used to

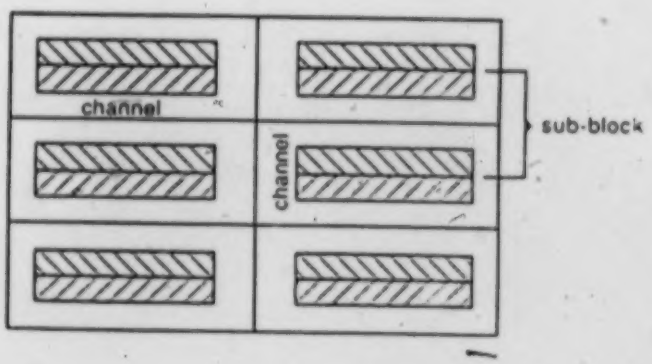


Figure 3.5 A standard-cell layout model

interconnect the associated sub-block nodes. The edges of the graph are assigned a weight which is a function of the geometric length of the interconnection and the estimated density of the channel. After all the nets have been assigned, the initial solution is improved by ripping up and re-routing selected nets.

The Timberwolf system [Sec84] uses double-entry cells with terminals on both the top and bottom edges of the cell. Each net is modelled by a graph with the edges weighted according to their Manhattan length. The global router finds a minimum length spanning tree for each net in turn. It then employs an iterative improvement step based on simulated annealing. This step essentially exchanges segments which are included in the spanning trees with those that are not in an attempt to reduce the total tracks used for the routing. The global routers in both the 'Polyplint' system [Anw85] and the 'ALPS2' system [Hsu85] take an approach similar to Timberwolf.

Global routers for gate-arrays can be adapted to standard-cell layouts because of the similarities in the two layout styles. Examples of gate-array global routers that use a layout model very similar to the one discussed in Section 2.1 are [Kan80, Ter82, Aos83, Wie84]. Since channel widths are fixed in gate-arrays, the objective of these gate-array global routers is to minimize the maximum channel density. In contrast, the objective in standard-cell routing is to minimize the sum of the channel densities.

Some deficiencies of all global routers considered above, and most existing global routers in general, are:

1. They use the graph theoretic approach almost exclusively. Such an approach tends to ignore much of the physical information of the routing environment which could be gainfully used in obtaining better solutions. For example, some constraints that the routing environment imposes could be used to an advantage in improving the quality of the routing.
2. They operate in a sequential fashion, ordering the nets in a particular way and decomposing them into a set of interconnections. A readily identifiable problem with such a sequential approach is that the quality of the solution depends on the order in which the nets are processed.
3. In routing each net, net-segments are selected for inclusion in the global routing based on 'local' conditions in their immediate neighbourhood. Because of the global nature of the routing problem, such a 'local optimization' could affect the quality of the routing.

A recent algorithm which attempted to overcome some of the above problems was developed at the University of Alberta. The algorithm is called the 'track-filling' global router [MaR85]. The track-filling algorithm cycles through the channels, building a track in each channel. A track is built by adding net-segments to it from left to right. Before a segment is added to a track, the algorithm builds a

set of mutually exclusive net-segments. Segments in this set must satisfy the following two criteria: (1) they lie entirely to the right of all the segments currently in the track and (2) their spans overlap one another. A segment is selected from the mutually exclusive set for filling a track on the basis of a classification and ranking scheme for net-segments.

One major drawback of the track-filling approach is that although the algorithm has at its disposal a ranking scheme that is able to adequately assess a net-segment (over all other net-segments on the chip) for inclusion in the global routing, it fails to use the scheme to its full potential. When a track is being filled, one segment from a mutually exclusive set *must* be selected for inclusion in the global routing. Therefore, net-segments are evaluated for selection only over segments in the immediate neighbourhood of the right-most segment in the track being filled.

Because the algorithms proposed in this thesis work use the above mentioned classification and ranking scheme, it is described below. The reader is referred to [MaR85] for a detailed discussion of the scheme.

Figure 3.6 illustrates examples of net-segments belonging to the various classes. Net-segments are broadly classified into two major categories: cross-channel and same-row net-segments. A cross-channel net-segment connects two terminals on neighbouring rows of a channel. A same row net-segment connects two terminals on the same row.

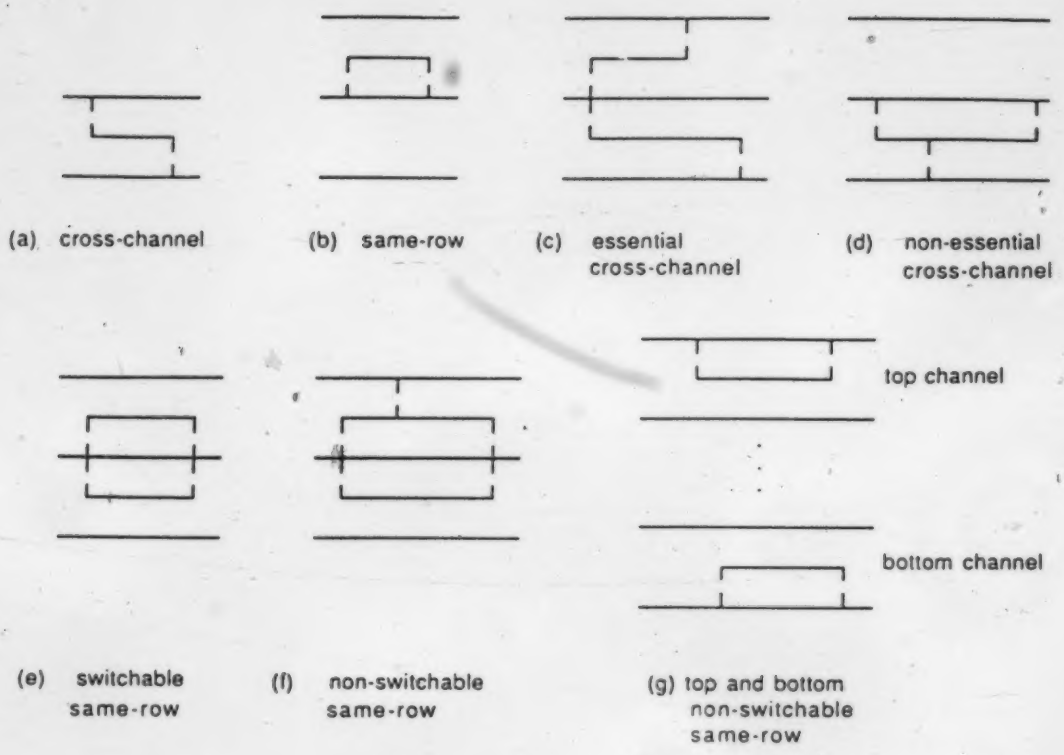


Figure 3.6 Classification of net-segments

Cross-channel net-segments are further classified into two types: essential and non-essential cross-channel net-segments. If a cross-channel net-segment in a channel is the only cross-channel net-segment for that net in that channel, then it is said to be an essential cross-channel net-segment. It is termed 'essential' because it must be part of the tree that connects the net. A cross-channel net-segment accompanied by other cross-channel segments for the same net in a channel is not essential, and is termed as a non-essential cross-channel net-segment. Same-row

net-segments are also classified into two types: **switchable** and **non-switchable** net-segments. If the two terminals of a same-row net-segment can be connected by another same-row net-segment in an adjacent channel, then the two same-row segments are termed as **switchable** same-row net-segments and they form a **switchable pair**. Often, the two terminals of a same-row net-segment are connected by more than one net-segment in the adjacent channel (Figure 3.6 (f)). Such a same-row segment is termed a **non-switchable** same-row net-segment. When the two terminals of a same-row segment lie on either the top or the bottom row, they cannot be connected in an adjacent channel because the areas above the top row and below the bottom row are not used for routing. Such same-row segments are therefore also termed as non-switchable net-segments. Further, if the two terminals of the non-switchable segment lie on the top row, the segment is termed as a **top non-switchable** net-segment; if the two terminals lie on the bottom row, the segment is termed as a **bottom non-switchable** net-segment.

Before proceeding with the description of the ranking scheme mentioned above, some terms used in the description will first be defined. The **local density** at a particular horizontal position, x in a channel is defined to be the number of nets which must cross, or start, or stop at the horizontal position x in the channel. The **segment density** of a net-segment is the maximum local density along the span of the net-segment. One measure used by the scheme to rank the

net-segments is based on the notion of **fullness**. The **local fullness** of a channel at a horizontal position x is defined to be

local fullness at x = local density at x/channel density.

The **segment fullness** of a net-segment is defined as

segment fullness = segment density/channel density

Since local density ≤ channel density, $0 \leq$ fullness ≤ 1. An additional parameter introduced is *FMAX*, the **maximum allowable fullness**. *FMAX* is a number between 0 and 1. Any region of a channel whose fullness is greater than or equal to the maximum allowable fullness is considered full.

All net-segments are ranked according to their (1) type (2) fullness and, where applicable (3) the fullness of the span of the net-segment in an adjacent channel. The ranking scheme categorizes the net-segments into the ten classes shown in Figure 3.7. CLASS 1 is ranked the highest and CLASS 10 the lowest. In Figure 3.7, f denotes the fullness of the net-segment s in its current channel and fa the fullness over the span of the net-segment in an adjacent channel. A net-segment is put into CLASSes 1 to 9 only if it is located in a channel interval which is not full (i.e. $f <$ *FMAX*). CLASSes 1 to 3 contain the switchable net-segments. They are ranked higher than the other types of segments because they can be re-assigned channels to effect a minimization of the sum of channel densities over all channels. Since essential net-segments must be selected eventually, they are ranked next to the switchable segments and are assigned to CLASS 4.

class			
1	$f_a \geq F_{MAX}$	full	Switchable
	$f < F_{MAX}$	not full	
2	$f_a < F_{MAX}$	more full	
	$f < f_a$		
	$f < F_{MAX}$	less full	
3	$f_a < F_{MAX}$		
	$f = f_a$	same fullness	
	$f < F_{MAX}$		
4	$f < F_{MAX}$	not full	Essential
5	$f < F_{MAX}$	not full	Non-essential
6	$f < F_{MAX}$	not full	Top and Bottom Non-switchable
7	$f_a \geq F_{MAX}$	full	Non-switchable
	$f < F_{MAX}$	not full	
8	$f_a < F_{MAX}$	more full	
	$f < f_a$		
	$f < F_{MAX}$	not full	
9	$f_a < F_{MAX}$		
	$f = f_a$	same fullness	
	$f < F_{MAX}$		
10	Others	$f \geq F_{MAX}$ or $f > f_a$	

Figure 3.7 Ranking of net-segments

The non-essential net-segments are ranked higher than the non-switchable segments based on the following observation. Figure 3.8 (a) shows a typical situation in which there is a choice between using a non-essential segment and a non-switchable one to connect up the terminals 1, 2 and 3 of a net. The net configuration of Fig. 3.8 (c) is preferable to (b) because a shorter total wire length is used and the local density is increased in one channel only. Among the non-switchable net-segments, the top and bottom non-switchable net-segments are ranked higher than other non-switchable segments. This is because the terminals of a top/bottom non-switchable segment cannot be connected in the area above/below the top/bottom channel, whereas terminals of a normal non-switchable segment may be connected in an adjacent channel.

We have discussed some of the limitations of existing global routers. We have also expressed the view that

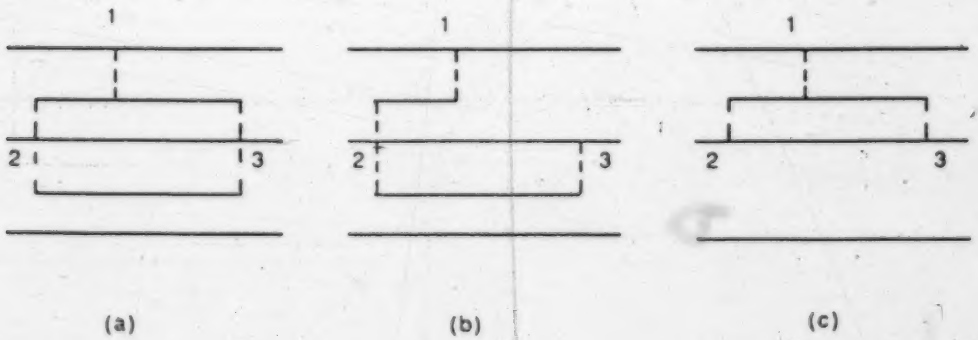


Figure 3.8 Choosing between a non-essential net-segment and a non-switchable net-segment

although a global routing phase can result in a better routing, not many global routers for standard-cell ICs have been reported. In the next chapter, we present a standard-cell global router that incorporates a new approach to global routing. The router avoids some of the problems of existing routers, dealing more effectively with the primary task of selecting net-segments for inclusion in the global routing. The following chapter discusses the proposed global routing algorithms.

Chapter 4

Algorithms for Global Routing

This chapter presents the new global router. The chapter begins by first establishing the motivation for the approach taken by the new global router. An analysis of net configurations in typical standard-cell circuits is presented, and based on the results of the analysis, a two phase approach to global routing is proposed. Three algorithms are presented for the first phase. The main features of each of the three algorithms are delineated and the algorithms are described. The chapter then proceeds to discuss the considerations in designing an algorithm for the second phase and concludes with a detailed discussion of the algorithm.

4.1 Motivation for the Approach taken to the Problem

With the aim of understanding the global routing problem in standard-cells better, several standard-cell circuits were examined to obtain statistics relating to net configurations in the circuits. Some of the circuits used for the purpose of the analysis have been designed, here, at the University. Two circuit designs have been procured from industries. A few circuit designs have been extracted from microprocessor data books.

The analysis of net configurations on these circuits indicated that 73% of the nets were either 2 or 3-terminal nets. The standard-cell routing environment constrains the number of possible net configurations for these nets. This is because net-segments (which constitute nets) connect only two adjacent terminals of a net in a channel. Therefore, the different ways in which the terminals of a net can be connected up by net-segments is limited. Figures 4.1 and 4.2 illustrate all the possible net configurations for 2 and 3-terminal nets. For 2-terminal nets there is very little flexibility in connecting the nets. The net-segment in Figure 4.1(a) would have to be connected as there is no alternate choice. We will refer to such a net-segment as a mandatory net-segment. (Equivalently, in the graph representation of a net, an edge incident on a vertex of degree 1 corresponds to a mandatory net-segment.) The net-segment in Figure 4.1(b) connecting two terminals on the same row is a switchable segment and can be routed either in the channel above the row or below the row. The two

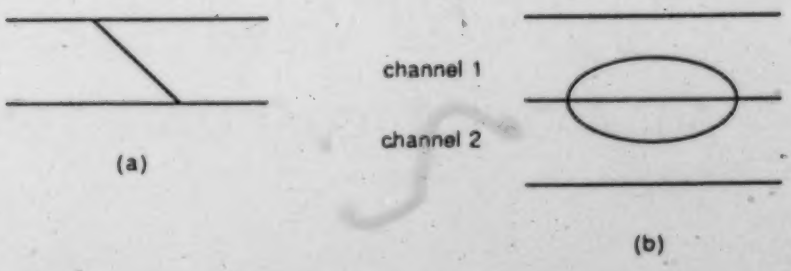


Figure 4.1 Net configurations for 2-terminal nets

switchable segments in Figure 4.1(b) form a pair. For 3-terminal nets, the configuration in Figure 4.2(a) only has mandatory segments while those in Figures 4.2(b) and (c) reduce to choosing a channel for the switchable segments. For the triad of Figure 4.2(d), two of the three net-segments need to be selected. Some further analysis was conducted on the net-segments of nets with 4 or more terminals. It was found that on the average 74% of the segments were either mandatory or switchable.

The above analysis of typical standard-cell circuits brought out an important point. Namely, it indicated that a large percentage of the nets were either already trees or the process of building a tree amounted to selecting one segment of a switchable pair or two of a triad. The results of the analysis formed the basis for the proposed approach.

In Chapter 3 some of the deficiencies of previous approaches to the problem had been pointed out. The sequential, minimum spanning tree type algorithms generate routing solutions which depend on the order in which the

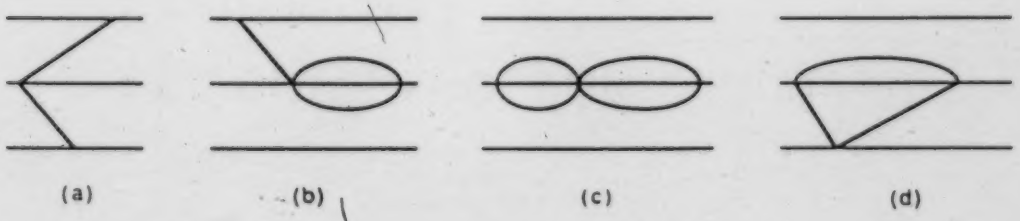


Figure 4.2 Net configurations for 3-terminal nets

nets are processed. Also, because they deal with only one net at a time, they attempt to optimize the figure of merit for that net only. These algorithms select net-segments on the basis of wiring congestion in the immediate neighbourhood of the segment in question. Though the 'track-filling' algorithm tries to overcome some of the above problems, it still resorts to evaluating net-segments for selection only over segments in the vicinity of the right-most segment in the track being filled.

The main motivation behind the proposed approach was the observation made that a majority of the nets were not that complicated. Because net-segments connect only two adjacent terminals of a net in a channel, and because introducing unnecessary feedthroughs is undesirable, the number of possible spanning trees for a net are constrained. Advantage can be taken of this observation in developing a heuristic for the global routing problem. Since, for each net the choice between net-segments is rather limited, it is felt that wherever a choice has to be made, it could be made with a more global view of wiring congestion on the chip rather than individually for each net. This will be accomplished by building more than one spanning tree at a time. Such an approach can not only address some of the problems outlined above but can also take advantage of the interaction between nets as net-segments are selected. Each net-segment is selected for inclusion in a global routing by comparing it with all the net-segments on the chip that have

not yet been selected. Also, some attempt is made to select net-segments by considering their effect on the real figure of merit, ΣD, rather than on the basis of local conditions within their respective channels.

The overall strategy adopted by the proposed algorithms is to focus on the two sub-problems: (1) choosing one segment of a switchable pair and (2) choosing two segments of a triad. Since the algorithms attempt a 'global optimization' in making the choice, it is anticipated that they would result in a better global routing.

4.2 A Two-Phase approach to Standard-Cell Global Routing

Global routing proceeds in two phases. Phase I, or the Net-segment Selection Phase, results in the selection of net-segments to form a spanning tree for each net. However, where a switchable segment is selected, the choice of a channel for it is relegated to Phase II. Phase II deals solely with choosing one segment from each pair of switchable segments, or, stated alternatively, with the assigning of switchable segments to channels. The main objective behind using a separate phase for assigning switchable segments to channels is that for large circuits they constitute between 40 to 60 percent of the total net-segments. It is thus felt that some benefit could accrue from trying to assign them optimally. Therefore, Phase II assigns these segments to channels considering their effect on the ultimate objective function, ΣD, rather than the

individual channel densities. Sections 4.2.1 and 4.2.2 below discuss the algorithms for Phase I and Phase II respectively.

4.3 Algorithms for Net-segment Selection

Three algorithms have been proposed for Phase I. They are Algorithm A, Algorithm B and the Select-Right algorithm. The main objective of each of these algorithms is to build a spanning tree for each net, processing several nets in parallel. This is achieved by first building for each channel the set of all net-segments in the channel. All these algorithms then proceed to select net-segments; each net-segment is evaluated for selection based on the ranking scheme discussed in Chapter 3. The ranking provides a basis for assessing (1) the relative importance of the net-segment to the spanning tree and (2) its suitability for selection over net-segments from other nets, not necessarily in its neighbourhood. Net-segments ranked higher are considered more 'desirable' for selection than those lower ranked. Each of the three algorithms is described in detail in the following sub-sections.

4.3.1 Algorithm A

Algorithm A selects net-segments for inclusion in the global routing based on two criteria: (1) their 'desirability' as assessed by the ranking scheme and (2) wiring congestion on the chip. By selecting segments first

in the least congested areas and then gradually extending to the more congested regions, the algorithm attempts to distribute wiring congestion between channels. The algorithm is given in Figure 4.3. This algorithm first builds for each channel a set, S_i, of all the net-segments in the channel and classifies them into one of the four types defined in the previous chapter, viz. (1) essential cross-channel (2) non-essential cross-channel (3) switchable same-row (4) non-switchable same-row. It then computes the densities of all the channels. For each channel i, the local densities along the channel are computed considering all the net-segments in S_i. The channel density, D_i is the maximum local density over the channel. The channels are ordered by increasing computed densities. It is expected that this ordering will enable the selection of net-segments in less congested channels first and could result in a better distribution of wiring congestion between channels.

The parameter *FMAX* is used to control the net-segment selection in algorithm A. As discussed in Chapter 3, *FMAX*, the maximum allowable fullness, is a number between 0 and 1. Any region of a channel whose fullness is greater than *FMAX* is considered full and the algorithm will not select net-segments in full regions. The following paragraph gives a brief explanation of how *FMAX* is used to permit the algorithm to select segments in less congested regions first.

```
Algorithm A - for net-segment selection
begin
  Build a set of net-segments, Si for each channel i
       and classify them according to their type;
  For each channel, initialize Ti to empty;
  Compute the density of each channel, Di;
  Order the channels by increasing densities;
  FMAX = the initial specified allowable maximum fullness;
  while not all Si's are empty do
    begin
      Rank the net-segments in Si for each channel;
      for RANK = 1 to 9 do
        begin
          for each channel in the sorted order do
            begin
              Select a net-segment from Si with
                    with rank = RANK;
              Add the net-segment to Ti, the set of
                  selected segments for the channel;
              Delete the net-segment from Si;
              Remove redundant net-segments;
            end;
        end;
      FMAX = FMAX + 0.1;
    end;
end;
```

Figure 4.3 Algorithm A

Starting with some initial specified value of *FMAX*, the algorithm proceeds to select net-segments from non-full regions. The algorithm cycles through the channels, selecting segments in increasing order of their Class. (It is to be noted that since Class 1 is the highest ranked and Class 10 the lowest, the selection proceeds in decreasing order of their 'desirability'). *FMAX* is incremented by 0.1 after every cycle through the channels. Therefore, some regions which were previously considered full are now not full and segments in these regions become enabled for selection. This allows the algorithm to select segments in regions with gradually increasing fullness.

As segments are selected in less congested regions first, they may help to make unselected segments in congested regions redundant. An unselected segment is said to be redundant if its two terminals are already connected by selected segments, i.e. there is already a signal path between its two terminals. When an unselected segment in S_i becomes redundant, it is deleted from S_i. Detection of redundant segments is facilitated by using 'global subnets' [MaR85] to record what terminals of a net are connected as segments are selected. Initially, each terminal is assigned to a separate global subnet. Let $G(t_i)$ be the global subnet that terminal t_i belongs to and $G(t_j)$ that to which terminal t_j belongs. When a net-segment (t_i, t_j) is selected, the subnets $G(t_i)$ and $G(t_j)$ are merged into a single subnet. An unselected segment in S_i becomes redundant if its two

terminals belong to the same global subnet.

As the algorithm cycles through the channels selecting segments, it builds spanning trees for all the nets. Each net-segment is selected by considering all other net-segments that have not yet been selected. The algorithm terminates when there are no segments remaining in any of the sets S_i.

4.3.2 Algorithm B

Algorithm B is a slightly less restricted form of Algorithm A. The algorithm is given in Figure 4.4. Without using the control parameter *FMAX* to facilitate selection of segments in less congested regions, the algorithm is permitted to select segments strictly on the ranking scheme. Essentially, since *FMAX* is given an initial value slightly higher than 1, all the net-segments over the entire chip become enabled for selection. Then as the algorithm cycles through the channels, it selects net-segments only on the basis of their ranks. The intention of experimenting with this form of algorithm is to determine:

(1) the effect of always selecting switchable segments over net-segments of other types, wherever such a choice exists. Because switchable segments are ranked the highest, this is essentially what the algorithm will do. Since switchable segments allow some flexibility in managing channel densities, the algorithm should yeild better solutions.

(2) the effect of always selecting non-essential segments

```
Algorithm B - for net-segment selection
begin
  Build a set of net-segments, Si for each channel i
     and classify them according to their type;
  For each channel, initialize Ti to empty;
  Compute the density of each channel, Di;
  FMAX = 1;
  RANK = 0;
  Rank all the net-segments in Si for each channel;
  Order the channels by increasing densities;
  while not all Si's are empty do
    begin
      RANK = RANK + 1;
      for each channel in the sorted order do
        begin
          Select a net-segment from Si
                with rank = RANK;
          Add the net-segment to Ti, the set of
                selected segments for the channel;
          Delete the net-segment from Si;
          Remove redundant net-segments;
        end;
    end;
end;
```

Figure 4.4 Algorithm B

over non-switchable ones, wherever such a choice exists. This will result in always selecting for the triad of Figure 4.2 (d) the two non-essential segments rather than a non-essential segment and a non-switchable one. As stated earlier, the advantage of using two non-essential segments to connect up the terminals of a triad is that a shorter total wire length is used and local densities are increased in one channel only. Therefore, ignoring any other considerations, could the strategy of always selecting the two non-essential segments of a triad be a good one?

In essence, Algorithm B selects as many switchable segments as it can and connects up all triads using two non-essential segments for each triad.

4.3.3 The Select-Right Algorithm

The central idea behind the Select-Right algorithm is to distribute wiring congestion between channels across a section of the chip while traversing the chip from left to right. The process can be viewed as a vertical scan line moving across the chip from left to right and selecting net-segments in its wake. Since the densities in the channels at a horizontal position just prior to the current position of the scan line are known, it is anticipated that this strategy could also come up with a reasonably good assignment of switchable segments between the channels.

The algorithm is given in Figure 4.5. The algorithm first builds a set containing all the net-segments on the

```
The Select-Right Algorithm - for net-segment selection

begin

  Build a list of all the terminals of all nets
      on the chip ordered by their horizontal
      position on the chip;

  while not end of the terminal list do

    begin

      Select net-segments incident on the terminal from
        the right according to the rules in Figure 4.6;

      Add the net-segments to the set of selected
        segments for their respective channels;

      Update the densities of the channels to which
        net-segments have been added above;

      Remove redundant net-segments;

    end;

end;
```

Figure 4.5 The Select-Right algorithm

chip and a list of all the terminals ordered by their horizontal position on the chip starting from the left. Then, the algorithm proceeds down the terminal list and at each terminal, selects net-segments that are incident upon the terminal from its right. The selection of net-segments is governed by a simple, 'greedy' heuristic consisting of a set of rules given in Figure 4.6. The figure illustrates four possible cases that could arise at a terminal, and specifies which segments are selected in each case. Essentially, the algorithm ensures that the current terminal

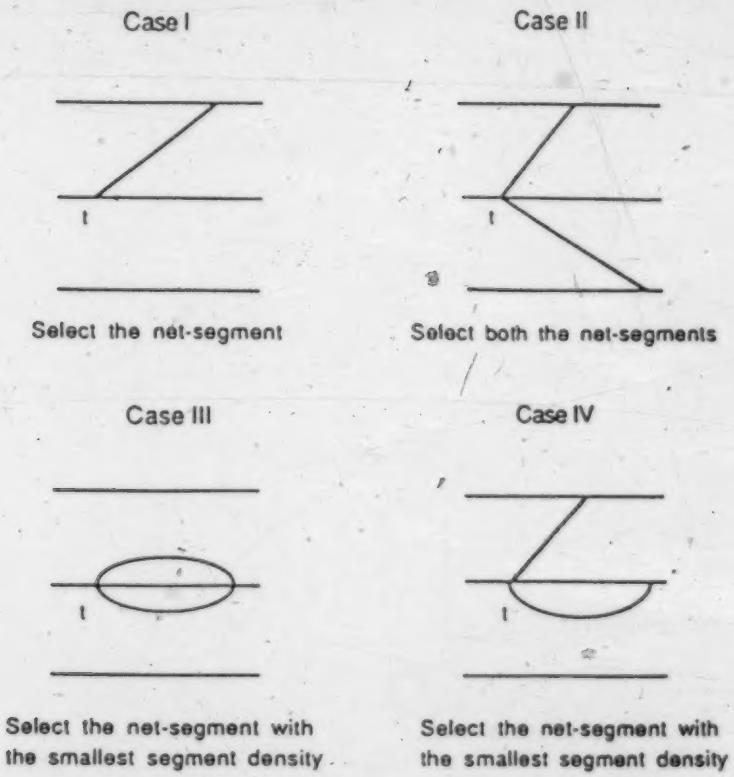


Figure 4.6 Rules for net-segment selection

of a net is connected up to at least one terminal of the net that is to its right. This ensures that when the algorithm terminates, spanning trees are built for all the nets on the chip.

4.4 An Algorithm for Assigning Switchable Net-Segments to Channels

This section discusses the algorithm for Phase II of the new global router. As mentioned earlier, Phase II deals specifically with the problem of assigning switchable

segments to channels. The section is organized as follows. First, the problem is stated as a combinatorial optimization problem. A currently popular approach for this class of problems, called Simulated Annealing, is then examined in some detail. Some of the drawbacks of this approach are pointed out followed by a discussion of how the proposed algorithm attempts to overcome some of these problems. The section then moves on to outlining the features that the proposed algorithm should incorporate, and concludes with a detailed description of the algorithm.

The problem of assignment of channels to switchable net-segments is a combinatorial optimization problem not unlike many problems that arise in CAD. It can be stated as follows: Given a set of n switchable net-segments, determine for each net-segment, whether the net-segment is to be routed in the channel above or below the row containing its two terminals, so as to minimize the sum of the channel densities over all channels. An exhaustive search for the optimum solution is infeasible since there are 2^n possible solutions. One of the goals of this research was to develop a heuristic algorithm to obtain a good solution to the above problem.

4.4.1 Motivation for the Design of the Algorithm

In recent literature, there have been numerous papers reporting significant success in using Simulated Annealing for a variety of optimization problems that arise in CAD

[Vec83, Fle85, Leo85]. Simulated Annealing was proposed by Kirkpatrick [Kir83] as an effective method for obtaining optimal or near-optimal solutions to combinatorial problems involving many degrees of freedom. The method is an extension of the Metropolis method [Met53] used to investigate properties of a collection of atoms in equilibrium at a given temperature. The Metropolis method provides a generalization of a procedure for incorporating controlled uphill steps in the search for a better solution to the optimization problem. Our interest in Simulated Annealing arose because of the Timberwolf Package [Sec84]. In Timberwolf, Simulated Annealing was used to obtain solutions to some layout problems, including the assignment of switchable segments to channels. Satisfactory results were reported. Wishing to accquire a better understanding of the capabilities and limitations of the method of Simulated Annealing in attempting to develop a better algorithm for the problem at hand, the method was studied in some detail.

4.4.2 Simulated Annealing

Most optimization problems that arise in the CAD area can be formulated as state space searches. The 'states' are correct or feasible solutions but not necessarily optimal solutions. Associated with each state is a cost function which measures the quality of the solution. The objective of an optimization is to find the state with the lowest cost. The Simulated Annealing method [Kir83] is suggested by an

analogy between the search for a state with the lowest cost in a combinatorial optimization problem and the process of finding low energy states in complex physical systems. An effective method for finding low energy states in a physical system, such as solids, is to heat the system up to some high temperature and then cool it slowly. This process, called annealing, lets the system settle into a low energy state, without getting trapped in higher lying local minima. In abstracting this process to optimization problems, the cost function of the system to be optimized is identified with the energy in the physical problem. Perturbations or new states of the system are generated by applying a set of moves to the system, and accepting or rejecting the perturbations based on the Metropolis criterion:

If $\Delta E \leq 0$ accept the perturbation

If $\Delta E > 0$ accept the perturbation with a probability $P(\Delta E) = e^{-\Delta E/T}$ where ΔE is the resulting change in the energy of the system due to the move and T is the temperature of the system.

As these perturbations are accepted or rejected at a fixed value of T, the system tends towards thermal equilibrium at that temperature. A typical annealing algorithm proceeds by starting the system at some high temperature. The system is allowed to approach equilibrium at that temperature. The temperature is then reduced and the system allowed to equilibrate again, and so on. The process is stopped at a temperature low enough that no more useful

improvement can be expected. The scheme used for cooling the system is known as the annealing schedule. The annealing schedule lists the sequence of decreasing temperatures and the desired number of accepted moves at each temperature. The basic Simulated Annealing algorithm is given in Figure 4.7.

The parameters T_0, α, the 'inner loop criterion' and the 'stopping criterion' are determined by the annealing schedule. T_0 is the initial temperature at which the procedure starts. The parameter α decides the sequence of decreasing temperatures. The criterion for determining whether the process has equilibrated at a particular temperature is specified by the 'inner loop criterion' and is generally given as the desired number of accepted moves at the temperature. The 'stopping criterion' specifies when the process is to be terminated. This could be either at the end of a specified temperature schedule or when not much improvement can be expected from continuing the process.

4.4.3 Some Observations

A Simulated Annealing algorithm similar to the one described in [Sec84] was implemented for the problem of assignment of channels to switchable net-segments. Details of the implemented algorithm are given in Appendix I. Experience with the algorithm indicated some major drawbacks with the method. Some of these have also been pointed out in [Nah85] for generated instances of the linear ordering

```
The Simulated Annealing Algorithm
begin
 S = initial state;  /* any feasible state */
 C(S) = cost function associated with state S;
 T = T0;  /* the initial temperature */
 while 'stopping criterion' is not satisfied do
   begin
     while 'inner loop criterion' not satisfied do
       begin
         generate a new state S';
         if ((C(S') < C(S))  or
                    (random < e^-(C(S') - C(S))/T))
               then S = S';
               /* random  is a uniformly generated
                  pseudo-random number in the range [0, 1] */
       end;
     T = α * T;
   end;
end;
```

Note: The parameters T_0, α, the 'inner loop criterion' and the 'stopping criterion' are specified by the annealing schedule.

Figure 4.7 The basic Simulated Annealing algorithm

problem. (The linear ordering problem consists of obtaining a linear ordering of the given n circuit elements, that minimizes the number of nets that cross between any pair of elements adjacent in this ordering.) First, the choice of an annealing schedule has a significant impact on the quality of the solutions produced. Some guidelines on selecting an annealing schedule are provided in [Whi84]. However, determining an optimal choice is not feasible since it depends not only on the particular problem being solved but also on the particular instance of the problem. Also, large amounts of computation time are required especially at low temperatures because many moves are rejected before each move to a different state.

Simulated Annealing is just one particular instance of a wider class of **Probabilistic Hill Climbing (PHC)** Algorithms that allow "hill climbing" moves [Rom84], i.e. they accept, with a certain probability, states that do not improve the cost function in order to climb out of local minima in the search for a global minima. In [Nah85], the performance of Simulated Annealing was compared with that of other algorithms belonging to the PHC class, for instances of the linear ordering problem. It was reported that on a set of randomly generated instances of the linear ordering problem, the other PHC algorithms often performed better than Simulated Annealing. The main motivation behind designing an algorithm for the problem at hand was to experiment with simple PHC algorithms along the lines of

those in [Nah85] and develop one that could perform comparably, or perhaps better, and avoid some of the problems of Simulated Annealing. Based on the results of the study in [Nah85] and the experience with the simulated annealing algorithm, some features that the proposed algorithm should incorporate were identified. They are:

1. The algorithm should maintain the approach taken earlier of 'order independancy' in routing nets i.e. without ordering the nets and then processing them in that order.
2. Since the algorithm should be much simpler to use than Simulated Annealing, with fewer parameters to select, the algorithm should accept uphill moves only if it has been unable over a sequence of attempts to find a good perturbation.
3. Given, in general, the constraint on computational resources and in view of the large amount of computing time that needs to be spent to obtain solutions with Simulated Annealing, it should be possible to stop the algorithm after a fixed amount of computing time.
4. Since PHC algorithms depend a great deal on the cost function, the proposed algorithm should use a cost function that is a good approximation of the real objective i.e. to minimize the sum of channel densities over all channels.

4.4.4 A PHC Algorithm for the Assignment of Channels to Switchable Net-segments

The algorithm is given in Figure 4.8. The algorithm will, in all future references to it, be called as the Simple Hill Climbing Heuristic (SHCH) Algorithm. The functions and procedures used in the algorithm are described below.

The SHCH Algorithm starts with the initial solution obtained after Phase I or the Net-Segment Selection Phase. A new state or perturbation is generated by randomly selecting a switchable net-segment and routing it on the opposite side of the row from its current position. If the perturbation is determined to be a good perturbation (we explain below the criteria for deciding a perturbation to be a good one), then it is accepted. Otherwise, it is rejected and the algorithm produces another perturbation. If the algorithm is unable to find a good perturbation over a sequence of attempts, it is presumed that a local minima has been reached, and the algorithm 'climbs out' by accepting a bad perturbation. The allowed length of the sequence of bad perturbations before one is accepted, is specified as an input parameter to the algorithm.

The procedure for determining whether a perturbation is a good one is illustrated with Figures 4.9 and 4.10. A new perturbation is generated by switching net-segment S from its current position in channel C_1 to channel C_2. Let D_1 and D_2 be the channel densities of C_1 and C_2 respectively. Let

```
Algorithm for Assignment of Channels to Switchable
                    Net-Segments - the SHCH Algorithm

begin

  S = initial state; /* any feasible state */
  L = allowed length of a sequence of bad
                                  perturbations;
  length = 0; /* the current length of the sequence
                          of bad perturbations; */
  while not out of computing time do

    begin

      generate a new state S';

      if (accept(S, S') = YES)

         then  S = S';

         /* accept returns YES if an acceptance
               criterion has been satisfied and
               returns NO otherwise */
    end;
end;


accept(S, S')
begin
  if it is a good perturbation
     then
          {
             length = 0;
             update the best solution found so far;
             return(YES);
          }
  else
      {
        length = length + 1;
        if (length > L)
           then
              {
                 length = 0;
                 return(YES);
              }
        else
              return(NO);
      }
end;
```

Figure 4.7 The SHCH Algorithm - for Phase II

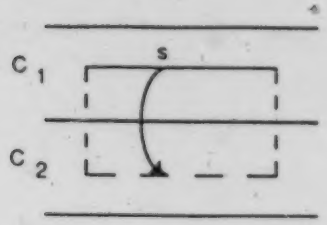


Figure 4.9 Generating a perturbation

d_1 be the segment density of the net-segment when in C_1 and d_2 when in C_2. One of the four cases illustrated in Figure 4.10 will arise. Perturbations that result in Case I are deemed good perturbations and are always accepted. Although such a perturbation may not always result in reducing the cost function, ΣD, it does eliminate one of the factors that keeps D_1 at its current value. A future pertubation then could result in reducing D_1. Perturbations that result in Cases II to IV are accepted only if there has been no good perturbation over a series of attempts.

The algorithm continues to search the solution space by generating new states and accepting or rejecting them in accordance with the 'acceptance criterion' described above, until it is out of computing time. Because the amount of computation involved in evaluating each state for acceptance is small, it allows the algorithm to consider a large number of states before terminating.

We have discussed above the objectives in designing the Two-Phase Global Router, and each of the four algorithms that constitute the global router, along with reasons why

Case I

$d_1 = D_1$ and $d_2 < D_2$

The perturbation could result in reducing D_1 by 1 without increasing D_2.

Case II

$d_1 < D_1$ and $d_2 < D_2$

The perturbation does not affect D_1 and D_2; it is therefore deemed unnecessary.

Case III

$d_1 = D_1$ and $d_2 = D_2$

The perturbation increases D_2 by 1; it may or may not reduce D_1 by 1.

Case IV

$d_1 < D_1$ and $d_2 = D_2$

The perturbation increases D_2 by 1; it does not affect D_1.

Figure 4.10 Results of a perturbation

better global routings are anticipated using the router. The global router has been implemented and the experiments conducted with the router are discussed in the following chapter.

Chapter 5

Experiments and Evaluation

This chapter discusses the experiments performed on the Two-Phase Global Router and the results of the experiments. The main objective of the experimentation was to evaluate the performance of each of the two phases seperately and of the global router as a whole. Section 5.1 describes the general environment for evaluating the global router as well as the test circuits used for the purpose. Sections 5.2 and 5.3 discuss the experiments conducted with the Phase I and Phase II algorithms respectively. Each of the sub-sections discuss details of the experiment conducted along with the results of the experiment and some observations on the results obtained.

5.1 Implementation and Environment for Evaluation

The algorithms described in the previous chapter have been implemented in the *C* programming language and presently form part of the U. of A. standard-cell design system. The system runs on a VAX 11/780 computer under UNIX[1] 4.3 BSD and consists of over 25,000 lines of source code. The input to the global router is a file produced by a placement program. The file describes the relative positions of the cells in the rows. The global router accesses the circuit database to

---

[1]UNIX is a trademark of Bell Laboratories

get the net-list of the circuit and the terminal positions. The global router generates a file which contains individual net-lists for each channel. This file is then used as input to the detailed router [MaR85]. Both the placement program and the detailed router are already part of the design system.

The global router was tested on 15 circuits with sizes ranging from about 100 cells to 1800 cells. Table 5.1 gives the statistics of the circuits. Circuits 10 and 11 are commercial circuits procured from industries. The rest have either been designed here at the University or are designs extracted from microprocessor data books. The circuits range from simple decoder, shift-register and adder/subtracter circuits to the more complex CRT controller circuit (Circuit 13) and the processor for floating point operations (Circuit 15). Since the objective of a global router is to minimize the total channel density, ΣD, we will use ΣD to evaluate the quality of the solutions. The following sections discuss the experiments conducted with the algorithms for each of the two phases and then those conducted to evaluate the overall performance of the Two-Phase Global Router.

5.2 Experiments with the Phase I Algorithms

Three algorithms were presented for the net-segment selection phase (Phase I): Algorithm A, Algorithm B and Select-Right. Experiments were conducted to examine whether any one of the three gave consistently better results. One

aim in designing the algorithms was to restrict the number of parameters used in the algorithms and avoid the inevitable 'tuning of the algorithm' to the parameters. The only two parameters that have been used are *FMAX* and a channel ordering. Some experiments were conducted to determine how these parameters should be used to yeild the best results.

5.2.1 Performance Comparison of the Algorithms for Net-segment Selection

Each of the 15 circuits was global-routed with the Two-Phase Global Router using each of the three algorithms for net-segment selection. The channels were processed in increasing order of their estimated densities. For Algorithm A, we need to input the value of *FMAX*, the initial maximum allowable fullness. We assume that on the average at least half of the net-segments in each channel will be selected to be routed in that channel. We therefore assign to FMAX an initial value of 0.5.

Since the algorithm for Phase II, which will be referred to as the SHCH algorithm, needs to know the amount of computing time available to it, the following strategy was adopted for the allotment of computer time. Circuits of size upto 400 cells were each allotted 2 minutes of CPU time. Circuits of size upto 800 cells were allotted 5 minutes each and those larger than 800 cells were given upto 10 minutes each of CPU time. For most of the circuits in

each of the three categories mentioned above, the final result was obtained in less than half the allotted time. Also, the decrease in the total channel density, ΣD, was most significant in the first minute of the run. However, the times were decided upon by observing the rate at which the values of ΣD fell. The time was allotted on the basis of the maximum time that the ΣD of a circuit in a particular category displayed a reasonable improvement rate.

The results of the experiment are presented in Table 5.2. They indicate that, in terms of the final total channel densities, slightly better results are obtained when Algorithm B is used for net-segment selection. A point worth noting is that although the total channel densities for all the three strategies are comparable after Phase I, the resulting improvement after Phase II is more in the case of Algorithm B as compared to the other two. We attribute this to Algorithm B always selecting non-essential cross-channel

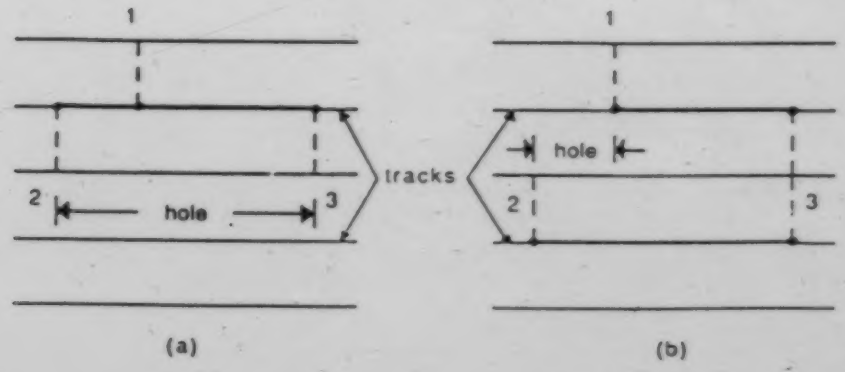


Figure 5.1 Choosing between a non-essential net-segment and a non-switchable net-segment

net-segments over non-switchable ones, wherever such a choice exists. Figure 5.1 shows that such a selection leaves wider 'holes' in the tracks than if one non-essential net-segment and one non-switchable one (Figure 5.1 (b)) were used. These wider 'holes' can be used more efficiently to move switchable net-segments between channels during Phase II, resulting in smaller total channel densities.

The running times for each of the three algorithms for Phase I is small compared to the time allotted for Phase II. Since the time for Phase II in the experiments conducted above is the same for the three algorithms, and depends on the size of the circuit, running times for all three algorithms can be considered comparable.

For each of the three algorithms, the processing time for a net-segment depends on a number of factors, notably (1) the type of the net-segment (2) its rank (3) the length of the net-segment (4) the number of net-segments in its net and (5) the number of net-segments in the channel in which it lies. An analysis of the algorithms is therefore difficult and will not provide a realistic indication of their actual running times. Therefore, their time complexities were empirically determined by noting the amount of computing time used by each algorithm to global-route the 15 test circuits. By plotting log_{10}(total number of net-segments) against log_{10}(time) and calculating the slope of the fitted straight line (Figures 5.2 and 5.3), the time complexity of Algorithm A was determined to be

about $O(E^{1.1})$ and that of Algorithm B to be $O(E^{1.2})$, where E is the total number of net-segments. Since the Select-Right algorithm performs global routing by processing each terminal of the terminal list, its time complexity was similarly determined by plotting log_{10}(total number of terminals) against log_{10}(time) (Figure 5.4), and found to be about $O(T^{1.5})$, where T is the total number of terminals.

5.2.2 Experiments with Channel Ordering and FMAX

For Algorithm A and B, it was proposed that the channels be processed in increasing order of their computed densities. It was anticipated that this would facilitate the selection of segments in less congested regions first and hence result in better solutions. Experiments were conducted to see if results supported this conjecture; channels were processed in ascending and descending order of their initial computed densities. Table 5.3 summarizes the results obtained for the 15 circuits.

There is no significant difference between the results obtained by the two ordering methods. Over the 15 circuits, neither method gave consistently better results. Though on the average, over the 15 circuits, results appear to support the proposed channel ordering scheme in the case of Algorithm B, for Algorithm A processing channels in descending order seems to be better.

To investigate how *FMAX* influences ΣD in the case of Algorithm A, experiments were conducted by varying the

initial value of *FMAX* between 0.1 and 1.0. Results indicated some small variation in ΣD for the different *FMAX*s. The deviations in ΣD were of the order of 0.1% or less and did not follow any pattern, therefore, tabulated results of the experiment have not been presented.

As explained in Chapter 4, the parameter *FMAX* is essentially used to select net-segments with smaller 'weights'. Therefore, when *FMAX* is assigned an initial value of 0.1 and then increased to 0.2, 0.3, ..., net-segments with weights less than *FMAX* get selected. On the other hand, when *FMAX* is assigned an initial value of 1.0, net-segments are *not* selected on the basis of their weights. The above results indicate that ΣD is not sensitive to *FMAX* and therefore to the weights of the net-segments. This can be attributed to the observation, made earlier in Chapter 4, that the standard-cell routing environment constrains the number of possible spanning trees for a net. The choice between net-segments to build a spanning tree for a net is therefore rather limited.

5.3 Experiments with the Phase II Algorithm

In Chapter 4 it was stated that the main reason for using a separate phase for assigning channels to switchable net-segments was that they constituted a large percentage of the net-segments. It was anticipated that better solutions could be obtained by expending special effort to assign them optimally. An examination of the results of Table 5.2 show

an improvement of upto 13% in the values of ΣD, with an average of 7% improvement over the 15 circuits. This demonstrates the effectiveness of Phase II in reducing ΣD.

The motivation for the design of the the SHCH algorithm was to develop a PHC algorithm that could perform comparably, or perhaps better than Simulated Annealing and avoid some of the problems of Simulated Annealing. Therefore, a Simulated Annealing algorithm was implemented so as to compare its performance with that of the SHCH algorithm.

The Simulated Annealing algorithm implemented was similar to that reported in [Sec84]. Extensive experiments were performed in arriving at an annealing schedule that gave the best results for the problem at hand. The major components of an annealing schedule are the initial temperature, T_o, the parameter α that decides the sequence of decreasing temperatures, the 'inner loop criterion' and the 'stopping criterion'. Finding optimal values of these parameters was time consuming. A deviation of up to 8% in the results was obtained with different combinations of the parameters. The results appeared most sensitive to α (which, in the experiments, was varied between 0.5 and 0.9) and T_o. T_o was varied between 0.6 and 10 resulting in the probability of initially accepting uphill steps varying between 0.2 and 0.9. It appeared impractical to determine the best combination of the values of the four parameters for each circuit. A significant effort was required in

arriving at a 'compromise' value of each of the four parameters for the 15 circuits. Details of the implemented algorithm and the annealing schedule are given in Appendix I.

To compare the performance of the SHCH algorithm with that of the Simulated Annealing algorithm, each of the two algorithms were used in turn as the Phase II algorithm to global-route the 15 circuits. In other words, one set of experimental results was generated by routing the 15 circuits with each of the three algorithms for Phase I followed by the Simulated Annealing algorithm. Another set was obtained by substituting the Simulated Annealing algorithm with the SHCH algorithm for Phase II. The second set of results was therefore generated by global-routing the 15 circuits with each of the three algorithms for Phase I followed by the SHCH algorithm for Phase II. To obtain a fair comparison, the SHCH algorithm and the Simulated Annealing algorithm were given an equal amount of computing time. The strategy for allotting time was the same as used in earlier experimentation, the allowed time depending on the size of the circuit.

The results of the experiment are given in Table 5.4. The solutions obtained by the SHCH algorithm for each of the 15 circuits are better than those obtained with the Simulated Annealing algorithm.

Because of the strict time allotment used in the experiment, the Simulated Annealing algorithm determined a

solution either at the end of the annealing schedule (i.e. a normal termination of the algorithm) or at the end of the allotted time, whichever came first. A natural question to ask is, in instances where the termination was not normal, could better results be obtained by letting the algorithm run its full course. This formed the basis of the next set of experiments. Using the Simulated Annealing algorithm as the Phase II algorithm, solutions were obtained for each instance at the end of the annealing schedule. Table 5.5 gives the results for each of the above three sets of data and the CPU time taken for each instance. Results show that the SHCH algorithm gives better results despite having restricted the time allowed to the algorithm to a maximum of 10 minutes. Running times for the larger circuits with the Simulated Annealing algorithm vary from 1/2 hour to up to 2 hours. Running times of this order were also reported in [Sec84]. In [Nah85], the authors had observed that for instances of the linear ordering problem, simple PHC algorithms often performed better than Simulated Annealing. The results of the above experiments with instances of the global routing problem appear to be in consensus with this observation.

The SHCH algorithm can also be used to improve global routing solutions produced by any other global router. In an experiment using the SHCH algorithm to improve solutions generated by the 'Track-filling' algorithm [MaR85], discussed in Chapter 3, an improvement of over 2% was

observed on the average.

5.4 Evaluation of the Two-Phase Global Router

To evaluate the Two-Phase Global Router, the performance of the global router was compared with that of other global routing algorithms. The algorithms used for performance comparison are already part of the U. of A. standard-cell design system and are discussed in [MaR85]. Each of the heuristics is briefly described below.

Since the approach taken by most global routers is to generate a minimum spanning tree for each net in sequence, the performance of the Two-Phase Global Router was compared with such a minimum spanning tree algorithm. Details of the algorithm are given in Appendix II which is reproduced from [MaR85]. Essentially, the algorithm orders the nets in increasing order of the number of terminals in the nets and finds for each net a minimum weight spanning tree. The edge weight is a function of the maximum local density along the channel interval between the two terminals connected by the edge.

The second heuristic used for performance comparison incorporates a method similar to the 'left-edge' algorithm [Hash71] to select net-segments for inclusion in the global routing. The algorithm processes the channels one at a time, building tracks in each channel. A track is filled from left to right selecting always the left-most net-segment of all net-segments not yet assigned to tracks.

In Chapter 3, we had discussed in detail a 'track-filling' global router [MaR85] which has been developed as part of the continuing research here at the University of Alberta. Experiments with the track-filling algorithm demonstrated good results. The research described in this thesis began by disputing the reasons for its good performance. It was because of this that an analysis of net configurations in standard-cell circuits was begun and the Two-Phase Global Router eventually proposed. The track-filling algorithm has therefore, naturally, been included among the heuristics for comparison.

As a matter of interest, one experiment was conducted to demonstrate the effectiveness of a global routing phase in improving final routing results. The circuits were routed without global routing, using a detailed router to route the channels one at a time from top to bottom.

The experimental results in terms of the total channel densities are presented in Table 5.6. For the experiment performed without using global routing, results indicate that total channel densities are over 12% higher than if global routing is used. The total channel densities produced with the 'left-edge' method were about 7% higher than those produced with the Two-Phase Global Router. Compared to the Minimum Spanning Tree algorithm, the Two-Phase Global Router results in smaller ΣD for each of the 15 circuits, with an average improvement of 4% over the 15 circuits. The intent of the performance comparison with the Track-filling

algorithm was to see whether a global router that primarily focused on the two sub-problems of selecting one segment of a switchable pair and that of selecting two segments of a triad, would lead to comparable results. Table 5.6 indicates that the overall performance of the Two-Phase Global Router is slightly better than that of the Track-filling algorithm.

It was indicated earlier that the Phase II algorithm of the Two-Phase Global Router was allowed to run for a maximum of 10 minutes for the larger circuits. These run-times appear higher than those reported for the Track-filling and Minimum Spanning Tree algorithms, which vary between 4 and 244 seconds of VAX 11/780 CPU time. However, for most of the circuits, the final result was obtained in less than half the allotted time.

A plot of the density profiles for one of the test circuits, before and after global routing is given in Figure 5.5. The figure shows that the final density profile is more flat than the initial one, indicating that wiring congestion is distributed fairly evenly between the channels. Figure 5.6 gives the density profiles after Phase I and II. It is seen that, for the top and bottom channels, the density profiles after Phase II follow the profiles after Phase I. This is perhaps because the net-segments that connect terminals on the top and bottom rows cannot be connected in an alternate channel. The profiles for the other three channels indicate that wiring congestion is distributed fairly well over the channels after Phase II.

The main objective in designing the Two-Phase Global Router was to show that better global routings could be obtained by focusing on the two sub-problems that arise in the global routing of standard-cell ICs and dealing with them effectively. The two sub-problems identified are the selection of one segment of a switchable pair and the selection of two segments of a triad. Results of the experimentation reported in this chapter show that the Two-Phase Global Router meets this objective. In the following chapter, we summarize this research work and present conclusions.

Table 5.1 Statistics of the test circuits

Circuit	Cells	Rows	Nets	%age of 2 and 3 terminal nets
1	86	6	76	63
2	88	5	93	80
3	116	6	119	77
4	162	7	156	83
5	218	6	217	71
6	258	6	272	79
7	306	7	306	61
8	392	8	408	68
9	400	8	442	74
10	446	8	523	73
11	578	10	612	71
12	631	11	570	66
13	802	12	766	69
14	1352	15	1266	79
15	1798	15	1728	81

Table 5.2 Experimental results obtained after Phases I and II using Select-Right, Algorithm A and Algorithm B as the Phase I algorithm for the Two-Phase Global Router

Circuit	Select-Right		Algorithm A		Algorithm B	
	Phase I	Phase II	Phase I	Phase II	Phase I	Phase II
	ΣD	ΣD	ΣD	ΣD	ΣD	ΣD
1	51	50	51	49	48	48
2	30	27	31	27	31	28
3	58	54	57	54	56	52
4	62	57	60	56	61	54
5	97	92	97	92	98	92
6	94	92	92	91	90	89
7	129	117	123	115	119	114
8	167	165	168	165	168	165
9	167	159	161	157	163	158
10	196	175	185	175	192	175
11	190	183	190	180	188	178
12	252	238	250	236	242	231
13	320	291	316	288	301	285
14	385	371	394	363	411	365
15	514	496	543	488	544	491
Total	2712	2567	2718	2536	2712	2525

Table 5.3 Experimental results obtained by varying the order in which channels are processed

Circuit size	Algorithm A		Algorihtm B	
	A	D	A	D
upto 400 cells	649	649	642	642
upto 800 cells	748	742	742	742
greater than 800 cells	1139	1138	1141	1155
Total	2536	2529	2525	2539

A = ascending order

D = descending order

Table 5.4 Experimental results comparing the SHCH algorithm with the Simulated Annealing algorithm, both algorithms being allowed the same CPU time per instance

Circuit	Select-Right		Algorithm A		Algorithm B	
	SHCH. ΣD	S. A. ΣD	SHCH. ΣD	S. A. ΣD	SHCH. ΣD	S. A. ΣD
1	50	50	49	49	48	48
2	27	27	27	28	28	28
3	54	54	54	54	52	52
4	57	58	57	57	54	55
5	92	93	92	93	92	93
6	92	92	90	90	89	90
7	117	123	115	118	114	118
8	165	167	165	167	165	166
9	159	160	157	158	158	158
10	175	182	175	180	175	181
11	183	186	178	182	178	181
12	238	245	232	237	231	236
13	291	300	285	292	285	293
14	371	378	368	374	365	369
15	496	506	485	497	491	504
Total	2567	2621	2529	2576	2525	2572

Table 5.5 Experimental results obtained using the Simulated Annealing algorithm

Circuit	Select-Right		Algorithm A		Algorithm B	
	ΣD	Time (mins)	ΣD	Time (mins)	ΣD	Time (mins)
1	50	0.3	49	0.3	48	0.3
2	27	1.4	28	1.4	28	1.4
3	54	1.7	54	1.5	52	1.7
4	58	3.1	57	3.3	55	3.3
5	92	5.0	92	4.9	93	5.4
6	92	4.3	90	4.7	89	5.0
7	119	9.0	117	9.2	117	9.3
8	167	6.9	165	7.4	166	7.2
9	160	9.0	157	9.1	158	8.5
10	179	25.7	177	27.5	178	27.1
11	183	18.4	178	18.9	180	19.2
12	241	22.5	236	23.5	233	23.1
13	294	33.3	286	34.8	290	35.3
14	372	65.3	370	66.7	368	63.0
15	502	138.1	489	137.0	493	129.0
Total	2590		2545		2548	

Table 5.6 Experimental results obtained with the Two-Phase Global Router, Track-Filling, Minimum Spanning Tree, and the Left-Edge algorithms, and with no global routing

Circuit	Two-Phase			Track-Fill	Min-span-tree	Left-edge	No global
	I	II	III				
	ΣD	ΣD	ΣD	ΣD	ΣD	ΣD	ΣD
1	50	49	48	48	48	51	50
2	27	27	28	28	30	30	31
3	54	54	52	56	55	58	64
4	57	57	54	58	60	61	66
5	92	92	92	91	95	96	99
6	92	90	89	93	92	97	102
7	117	115	114	119	120	128	136
8	165	165	165	167	172	169	172
9	159	157	158	160	164	166	170
10	175	175	175	173	184	188	200
11	183	178	178	181	186	194	209
12	238	232	231	238	240	250	256
13	291	285	285	291	299	320	324
14	371	368	365	362	381	392	413
15	496	485	491	479	508	509	542
Total	2567	2529	2525	2544	2634	2709	2834

I, II and III refer to using Select-Right, Algorithm A and Algorithm B resp. for Phase I.

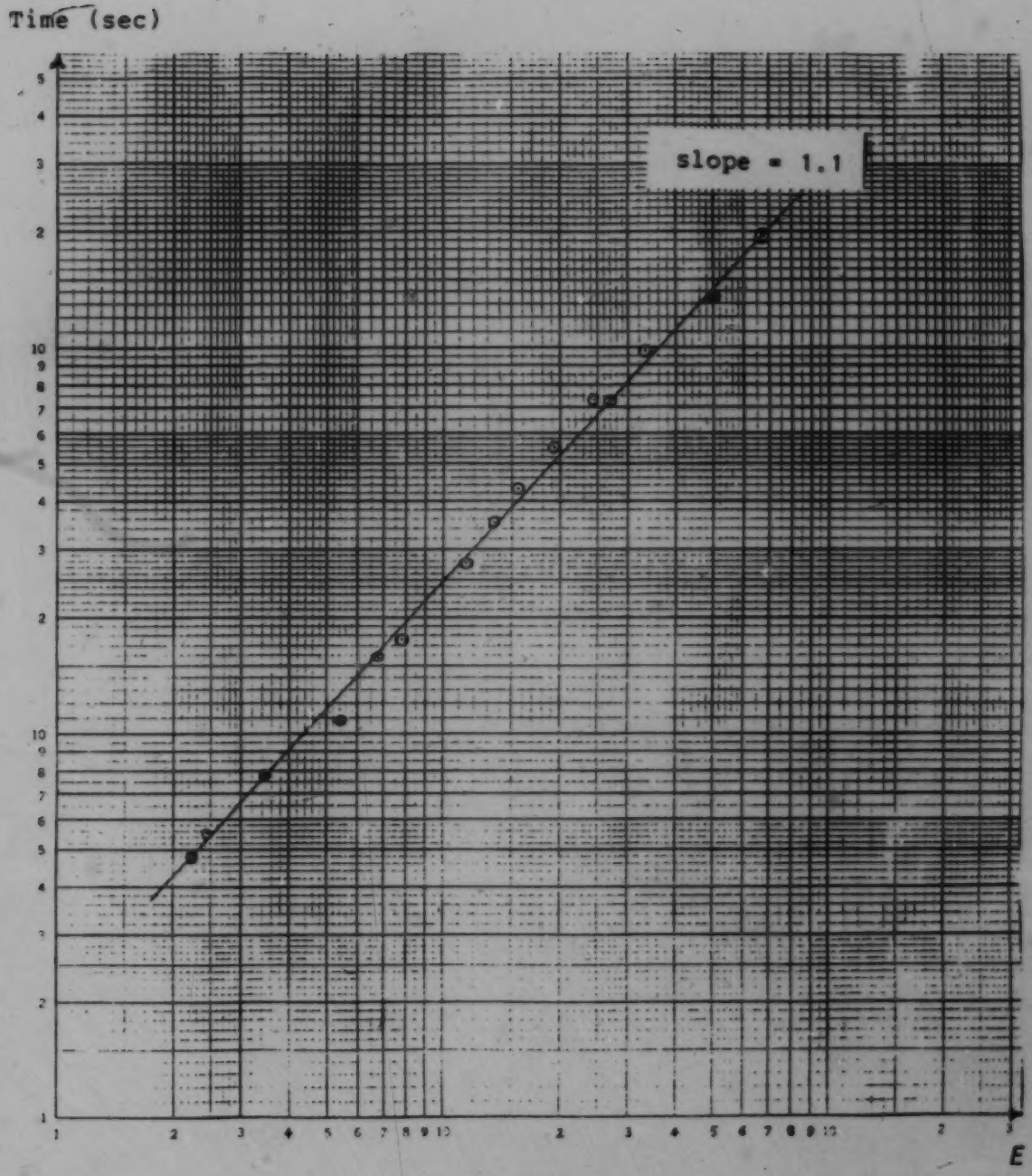


Figure 5.2 log_{10}(total number of net-segments) vs log_{10}(time) for Algorithm A

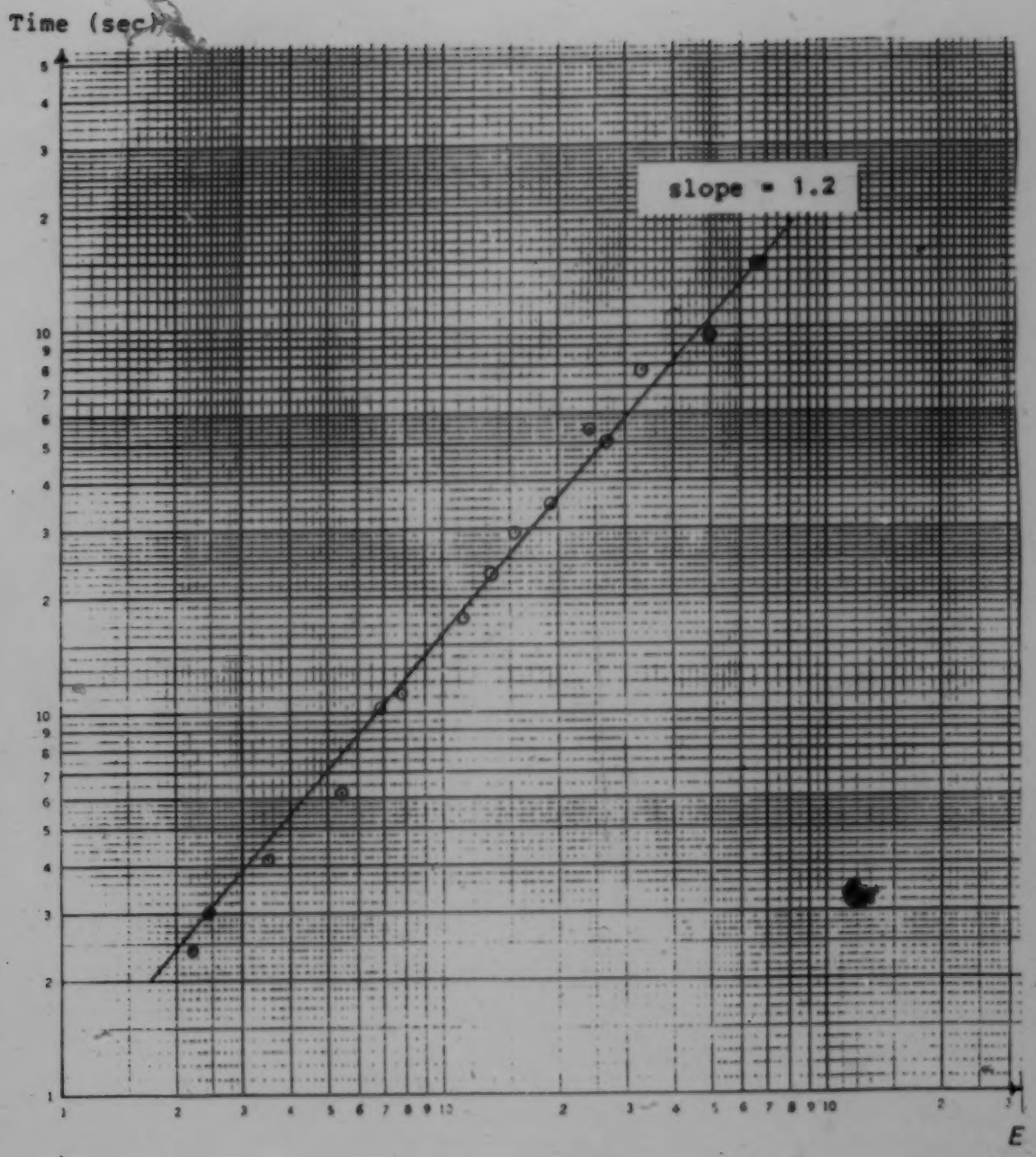


Figure 5.3 log_{10}(total number of net-segments) vs log_{10}(time) for Algorithm B

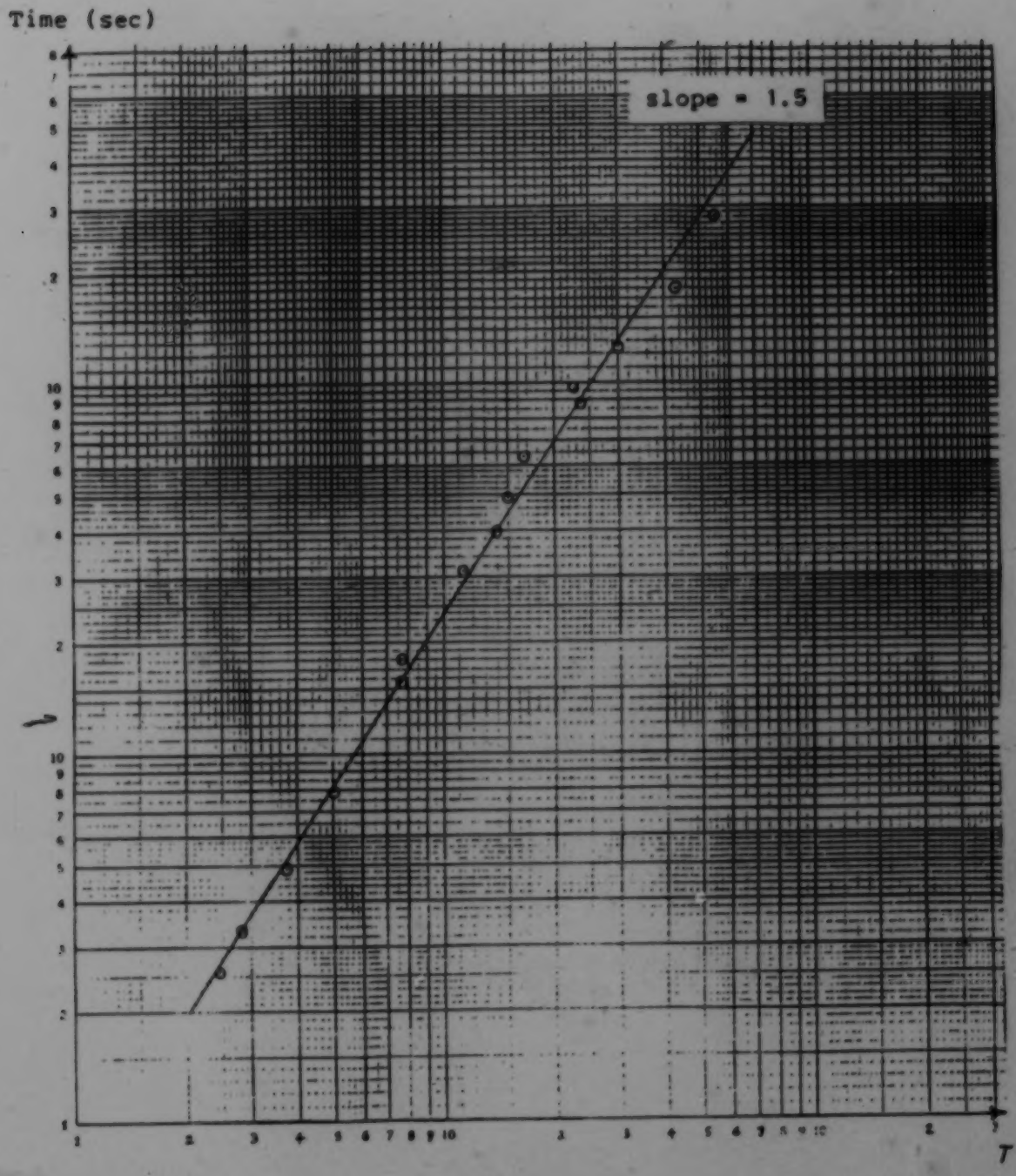


Figure 5.4 log_{10}(total number of terminals) vs log_{10}(time) for the Select-Right algorithm

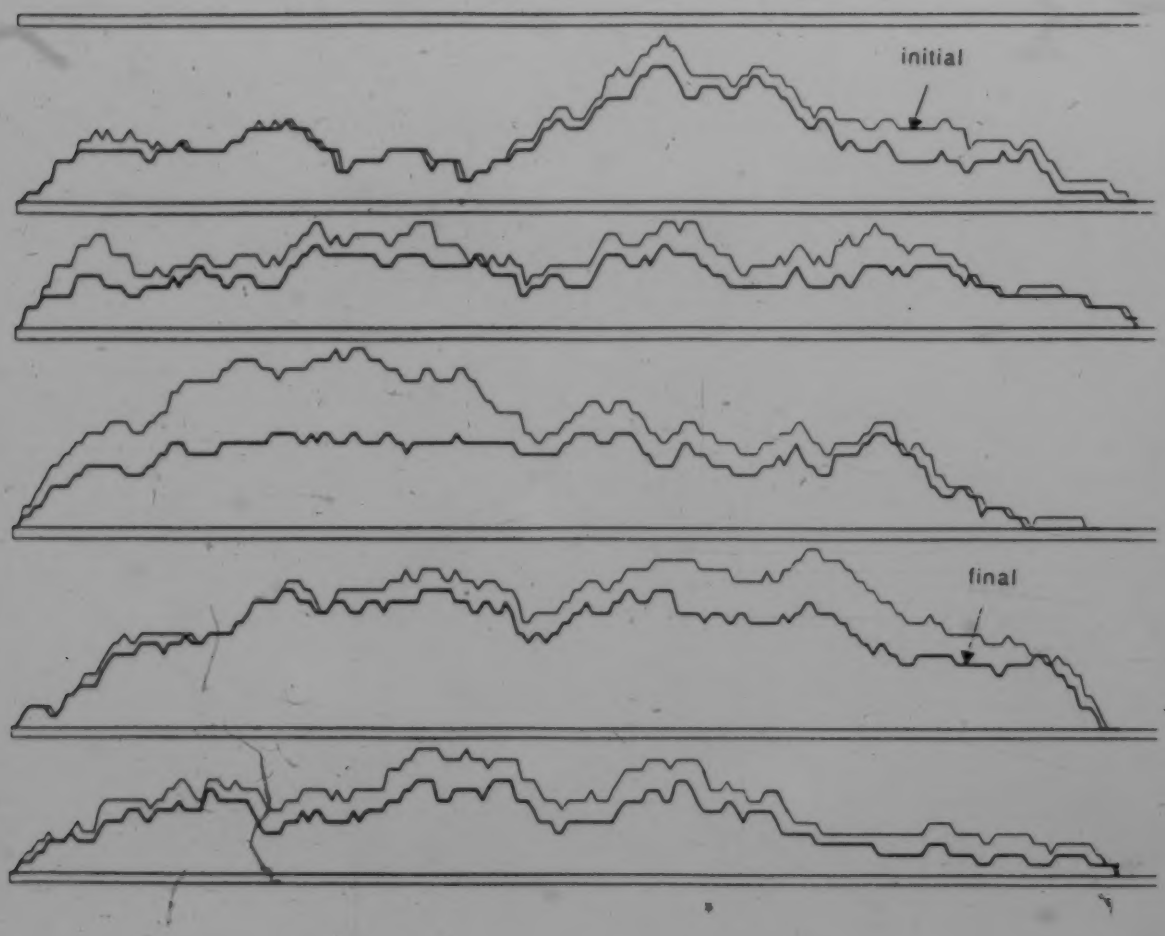


Figure 5.5 Initial and final density profiles

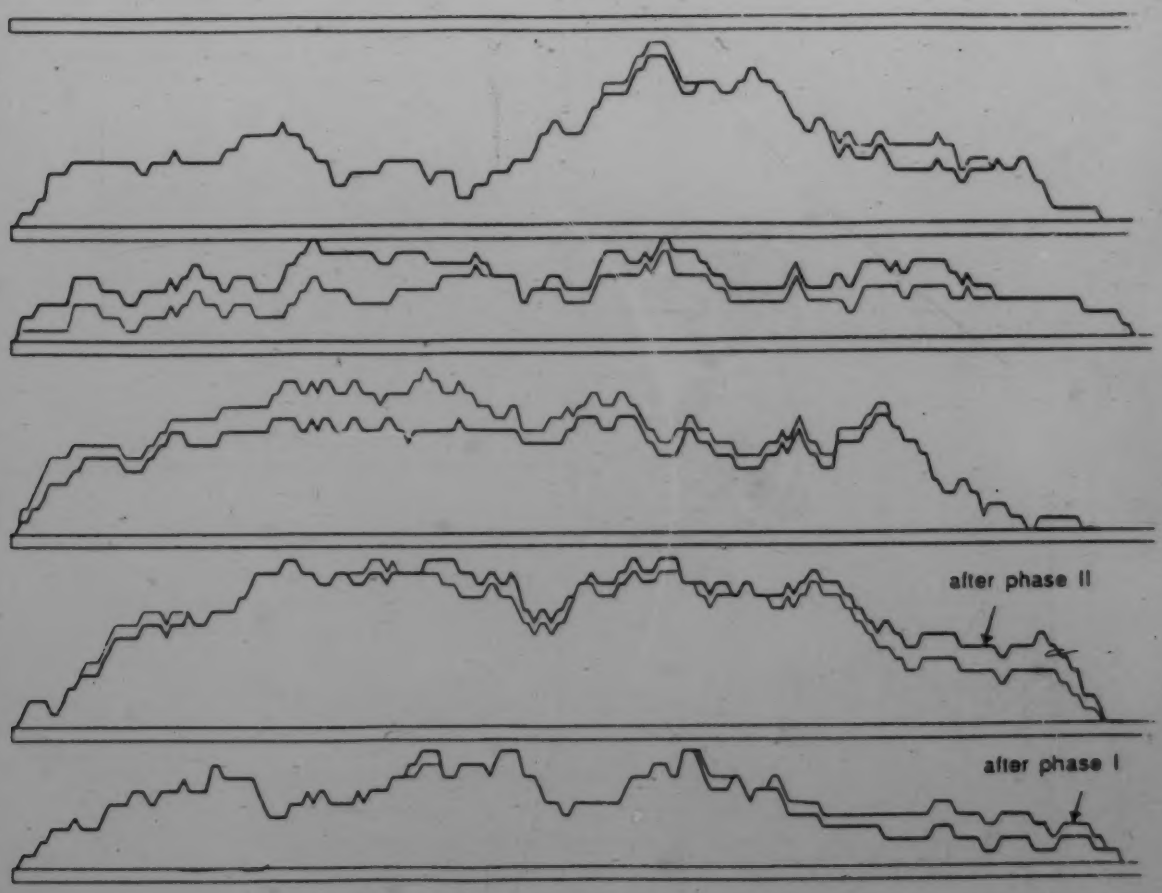


Figure 5.6 Density profiles after Phases I and II

Chapter 6

Conclusions and Directions for Future Work

The work presented in this thesis is part of the on-going research, here at the University of Alberta, into developing efficient algorithms for standard-cell layout. The major contribution of this thesis is a global router for standard-cell layouts that incorporates a new approach to the global routing problem.

Early in our research work we noted that few standard-cell layout systems employed a global router in routing the circuits. We felt that introducing a global routing phase could result in a better routing. Results of experiments conducted in this thesis work clearly demonstrate the effectiveness of a global routing phase prior to detailed routing. Over the 15 circuits used for performance comparison, total channel densities were, on the average, about 12% higher than if global routing is used.

Our initial analysis on net configurations of actual standard-cell circuits suggested that a rather different approach from the more conventional sequential, minimum spanning tree type global routers was needed. The analysis indicated that most of the nets were either already trees or the choice was limited to selecting either one segment of a switchable pair or two of a triad. The approach we have followed in our Two-Phase Global Router is that wherever a

choice is to be made between net-segments for a net, it is made with a global view of wiring congestion on the chip rather than individually for each net. As a consequence, the set of net-segments to finally connect up each net is obtained processing nets in parallel. We, therefore, avoid the 'net ordering' problem inherent in global routers that process nets sequentially. Results of experiments conducted to evaluate the performance of our global router show that for each of the 15 circuits used for performance comparison, our router resulted in smaller total channel densities as compared to a spanning tree type algorithm, with an average improvement of 4% over the 15 circuits.

We have shown in this thesis an alternative and viable approach to the global routing problem in standard-cell layouts. We have not, however, been able to explore some issues to our full satisfaction. Results of our experiments with the Phase II algorithm indicate that significant improvements can be obtained by trying to optimally re-assign switchable net-segments to channels. The possibility of improvements resulting from relegating the selection of net-segments of a triad to Phase II or perhaps a later phase has not adequately been investigated. The present algorithm limits itself to one of two possible selection rules: (1) selecting the two non-essential cross-channel net-segments or, (2) selecting the best two segments. However, the 'best' is decided upon by 'local' conditions in their respective channels. Perhaps Phase II

should not just concentrate on the problem of assignment of channels to switchables but also the selection of segments from a triad, to improve routing solutions.

One extension that merits some further work is incorporating into Phase II, or perhaps a later phase, the moving of non-switchable net-segments between channels. Each such move could be based on a cost function that represents the net benefit of such a move. Switching a non-switchable segment to an adjacent channel involves the addition of one feed-through cell in a row. Figure 6.1 shows an example of such a switch. The non-switchable net-segment of net 3 is moved to channel 1 thereby reducing the channel density of channel 2 from 2 to 1. If such a switch could reduce the number of tracks, what would be the resulting increase in the length of the rows and could it result in a net reduction in chip area?

The Select-Right algorithm, we feel, has potential for further development. The algorithm incorporates one pass of

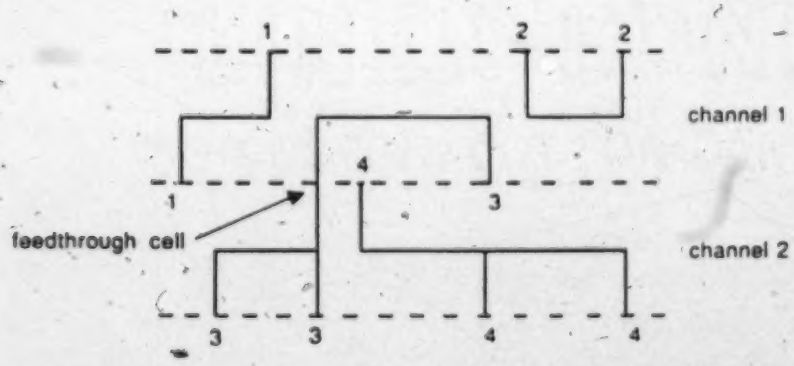


Figure 6.1 Moving a non-switchable net-segment

a cursor across the chip from left to right and selects net-segments in its path. A second pass could be introduced to perform a detouring of net-segments, (or sections of a net-segment) moving them away from congested regions to improve the routing solution. Figure 6.2 shows an example of such a detouring. Switching of switchable net-segments between channels is a particular case of detouring. However, such a switching need not be restricted to switchable segments. Also, it need not be limited to the adjacent channels if improvements can result from the detouring. For the system presented in [Ter82], the detouring of net-segments is performed manually to improve the solution generated by the router. Introducing a second pass in the Select-Right algorithm could enable such a detouring to be performed without manual intervention and facilitate better track utilization in the channels.

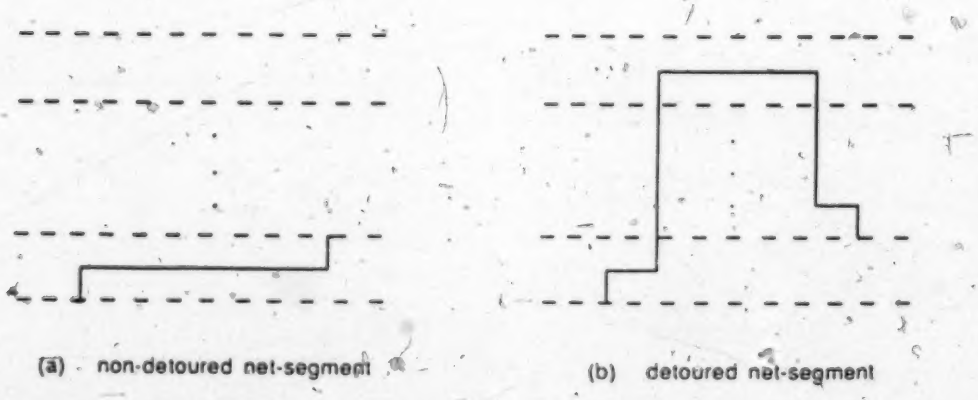


Figure 6.2 Detouring of a net-segment

Bibliography

[Anw85] H. Anway, G. Farnham, R. Reid, "Plint Layout System for VLSI Chips", *Proceedings of the 22nd Design Automation Conference*, 1985, pp. 449-452.

[Aos83] K. Aoshima, E. S. Kuh, "Multi-Channel Optimization in Gate-Array LSI Layout", *Proceedings of the 1983 IEEE International Symposium on Circuits and Systems*, 1983, pp. 1005-1008.

[Bur85] M. Burstein, "Channel Routing", *IBM Research Report RC 10973*, 1985.

[Fle85] H. Fleisher, J. Giraldi, D. B. Martin, et al, "Simulated Annealing as a tool for Logic Optimization in a CAD Environment", *Proceedings of the 1985 IEEE International Conference on Computer Design*, 1985, pp. 203-205.

[Hash71] A. Hashimoto, J. Stevens, "Wire Routing by Optimal Channel Assignment Within Large Apertures", *Proceedings of the 8th Design Automation Workshop*, 1971, pp. 155-169.

[Hig69] D. W. Hightower, "A Solution to the Line-Routing Problem on the Continuous Plane", *Proceedings of the 6th Design Automation Workshop*, 1969, pp. 1-24.

[Hsu85] C. P. Hsu, B. N. Tien, K. Chow, R. A. Perry, et al, "ALPS2, A Standard Cell Layout System for Double-Layer Metal Technology", *Proceedings of the 22nd Design Automation Conference*, 1985, pp. 443-447.

[Joh82] D. S. Johnson, "The NP-Completeness Column: an Ongoing Guide", *Journal of Algorithms*, Vol. 3, No. 4, 1982, pp. 381-395.

[Kan80] H. Kanada, K. Okazaki et al, "Channel-Order Router, A New Routing Technique for a Masterslice LSI", *Journal of Digital Systems*, Vol. 4, 1980, pp. 427-441.

[Kim83] S. Kimura, N. Kubo, T. Chiba, I. Nishioka, An Automatic Routing Scheme For General Cell LSI", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-2, No. 4, Oct. 1983, pp. 285-292.

[Kir82] S. Kirkpatrick, C. D. Gelatt, Jr., M. P. Vecchi, "Optimization by Simulated Annealing", *Science*, Vol. 220, May 1983, pp. 671-680.

[Kol77] K. W. Koler, U. Lauther, "The Siemens-AVESTA-System for Computer-Aided-Design of MOS-Standard Cell Circuits", *Proceedings of the 14th Design Automation Conference*, 1977, pp. 153-157.

[Kru56] J. B. Kruskal, Jr, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Porblem", *Proc. American Mathematical Society*, Vol. 7, No. 1, 1956, pp. 48-50.

[LaP80] A. S. LaPaugh, "Algorithms for Integrated Circuit Layout: An Analytic Approach", Ph.D. Dissertation, MIT Lab. Computer Science, Nov. 1980.

[Lee61] C. Y. Lee, "An Algorithm For Path Connection and its Applications", *IRE Transactions on Electronic Computers*, Vol. EC-10, 1961, pp. 346-365.

[Leo85] H. W. Leong, D. F. Wong, C. L. Liu, "A Simulated-Annealing Channel Router", *Proceedings of the 1985 IEEE International Conference on Computer Design*, 1985, pp. 226-228.

[MaR85] R. Ma, "Standard-Cell Routing", MSc. Thesis, University of Alberta, 1985.

[Meh84] S. Mehta, B. Kirk, M. Ng, R. Babbar, "CIPAR - A Complete Correct-by-Construction Placement and Routing System", *Proceedings of the 1984 IEEE Custom IC Conference*, 1984, pp. 117-121.

[Met53] N. Metropolis, A. Rosenbluth, A. Teller, E. Teller, "Equation of State Calculations by Fast Computing Machines", *Journal of Chemical Physics*, Vol. 21, No. 6, 1953, pp. 1087-1092.

[Mik68] K. Mikami, K. Tabuchi, "A Computer Program for Optimal Routing of Printed Circuit Connectors", *Proceedings of the IFIPS*, Vol. H47, 1968, pp. 1475-1478.

[Nah85] S. Nahar, S. Sahni, E. Shragowitz, "Experiments with Simulated Annealing", *Proceedings of the 22nd Design Automation Conference*, 1985, pp. 748-752.

[Riv82] R. L. Rivest, C. M. Fiduccia, "A Greedy Channel Router", *Proceedings of the 19th Design Automation Conference*, 1982, pp. 418-424.

[Rom84] F. Romeo, C. Sechen, A. Sangiovanni-Vincentelli, "Research on Simulated Annealing at Berkely", *Proceedings of the 1984 IEEE International Conference on Computer Design*, 1984, pp. 652-657.

[Rub74] F. Rubin, "The Lee Connection Algorithm", *IEEE Transactions on Computers* Vol. C-23, 1974, pp. 907-914.

[Sec84] C. Sechen, A. Sangiovanni-Vincentelli, "The TimberWolf Placement and Routing Package", *Proceedings of the 1984 IEEE Custom IC Conference*, 1984, pp. 522-527.

[Sou78] J. Soukup, "Fast Maze Router", *Proceedings of the 15th Design Automation Conference*, 1978, pp. 100-102.

[Szy85] T. G. Szymanski, "Dogleg Channel Routing is NP-Complete", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-4, No. 1, Jan. 1985, pp. 31-41.

[Ter82] M. Terai, H. Kanada, K. Sato, T. Yahara, "A Consideration of the Number of Horizontal Grids used in Routing a Masterslice Layout", *Proceedings of the 19th Design Automation Conference*, 1982, pp. 121-128.

[Tin83] B. S. Ting, B. N. Tien, "Routing Techniques for Gate-Arrays", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-2, No. 4, Oct. 1983, pp. 301-312.

[Vec83] M. Vecchi, S. Kirkpatrick, "Global Wiring by Simulated Annealing", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-2, No. 4, Oct. 83, pp. 215-222.

[Whi84] S. White, "Concepts of Scale in Simulated Annealing", *Proceedings of the 1984 IEEE International Conference on Computer Design*, 1984, pp. 646-651.

[Wie84] M. Wiesel, "Loose Routing For Gate Arrays", *Proceedings of the 1984 IEEE International Symposium on Circuits and Systems*, 1984, pp. 444-448.

[Yos82] T. Yoshimura, E. S. Kuh, "Efficient Algorithms For Channel Routing", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. CAD-1, No. 1, Jan. 1982, pp. 25-35.

Appendix I - A Simulated Annealing Algorithm for Assigning Switchable Net-segments to Channels

The basic Simulated Annealing algorithm is given in Figure 4.7. In adapting it to the problem of assigning switchable net-segments to channels, the algorithm starts with the solution obtained after the Net-segment Selection Phase as the initial state, S. The channel densities of all the channels are determined. The sum of the channel densities over all channels, ΣD, is the initial value of the cost function, $C(S)$. A new state, S' is generated by randomly selecting a switchable net-segment and routing it on the opposite side of the row from its current position. The cost function, $C(S')$ is evaluated for the new state. Let $\Delta C = C(S') - C(S)$. As a result of the new state, the cost function either increases by 1, decreases by 1, or remains the same, i.e. $\Delta C = 1, -1$, or 0 respectively. New states with $\Delta C = -1$ or 0 are always accepted. For states with $\Delta C = 1$, the parameter, T determines whether they will be accepted. The states are accepted if random $< e^{-1/T}$, where random is a uniformly generated number in the range [0, 1]. Initially, $T = T_0$.

After experimenting with values of T_0 between 0.6 and 10 (resulting in the probability of initially accepting uphill steps varying between 0.2 and 0.9), T_0 was set to 0.621, with $e^{-1/T_0} = 0.2$.

The 'inner loop criterion' is implemented by specifying the number of new states to be generated for each stage of

the annealing process. This number is specified as a multiple of the number of switchable net-segments, *N*, in the circuit under consideration. It has been assigned a value of 30 * *N*.

The 'stopping criterion' is implemented by specifying the number of stages, or temperatures to be considered in the annealing process. After experimenting with a 6 temperature schedule and a 10 temperature one, a 6 temperature schedule was decided on.

The parameter, α which determines the sequence of decreasing temperatures, has been assigned a value of 0.5, after some experimentation with values of α ranging between 0.5 and 0.9.

Appendix II - A Minimum-Spanning-Tree Global Router

The tree-type global routing algorithm we implemented is similar to the algorithms of [Aos83], [Sec84] and [Wie84]. Basically, the algorithm orders the nets by increasing number of terminals in the nets. In the sorted order, the algorithm finds, for each net, a minimum weight spanning tree of canonical wire segments which connect the net. The edge weight is $1/(2^{**}(D-d))$, where D=channel density, and d=maximum local density along the channel interval between the two terminals connected by the edge.

The minimum spanning tree is found using Kruskal's algorithm [Kru56]. Kruskal's algorithm takes $O(n\log_2 n)$ time to find the minimum spanning tree for a graph with n edges. Let N=number of nets, E=total number of canonical wire segments, e=average number of segments per net, i.e., $e=E/N$=average number of edges in the graph for each net. Therefore, on average, the time required to compute the minimum spanning tree for N nets each with e edges is about $O(Ne\log_2 e)=O(E\log_2 e)$ since $Ne=E$.